UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Evaluación de Algoritmos de Seguimiento de Objectos

-PROYECTO FIN DE CARRERA-

Mónica Lozano Cruz
Febrero 2012

Evaluación de Algoritmos de Seguimiento de Objectos

Autor: Mónica Lozano Cruz
Supervisor: Juan Carlos San Miguel Avedillo
Ponente: José María Martínez

email: Monica.Lozano@estudiante.uam.es, Juancarlos.Sanmiguel@uam.es
Josem.Martinez@uam.es

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Febrero 2012

Abstract

The main objective of this master thesis is the evaluation of video tracking algorithms. A protocol is designed for such evaluation task including the following aspects: accuracy, robustness to initialization errors, computational cost and optimum parametrization. In order to support the evaluation, a dataset is defined considering the most common problems in video tracking. Finally, the proposed protocol is tested on representative approaches of the state-of-the-art.

Resumen

El objetivo principal de este PFC es la evaluación de diferentes algoritmos de seguimiento de objetos. Para realizar dicha evaluación se ha diseñado un protocolo que evaluará los siguientes aspectos: precisión, robustez a errores de inicializacion, coste computacional y parametrización óptima. Por otro lado, se ha diseñado y creado un dataset completo que incluye algunos de los problemas más comunes en seguimiento de objetos. Finalmente, el protocolo se utiliza para evaluar un conjunto de algoritmos previamente seleccionados.

Keywords

Video analysis, video object tracking, performance evaluation, evaluation metrics, evaluation protocol, dataset design.

Agradecimientos

Comienzo estos agradecimientos nombrado a la persona que más me ha ayudado en la realización de este proyecto: mi tutor Juan Carlos San Miguel. En todos estos meses ha mostrado un apoyo sin el que habría sido muy difícil mantener esa motivación tan alta que me ha inculcado. Siempre dispuesto a resolver dudas, comentar soluciones o limar detalles para conseguir el resultado más pulido y perfecto posible. Agradecer también a José María Martínez por escucharme y ofrecerme soluciones cuando decidí comenzar este proyecto y a Jesús Bescós, que con su manera de enseñar tan cercana y su puerta del despacho siempre abierta ha hecho durante todos estos años que cada reto fuera un poquito más llevadero. Gracias también a todos los compañeros del VPULab, que me dieron un sitio perfecto donde concentrarme y llegar con ganas de trabajar, y que siempre se mostraron dispuestos a ayudar en lo que hiciera falta.

Muchas gracias también a todos mis amigos que me acompañaron durante los años que sin duda recordaré como de los más intensos de mi vida. Gracias Guss, Peter, Pablo, Kiko y Chus por todas esas noches, esos Guardamares, esos SanMis, esos días de prácticas hasta las mil de los que salíamos sin parar de reirnos y decir tonterías. Gracias a mis niñas, que me han enseñado tanto, muchas veces sin saberlo. Gracias Ele, por decir siempre lo que necesito oír aunque no quisiera hacerlo. Gracias Esther, por ser una de las personas que me enseñó a hablar de verdad. Gracias Sonso, por haber sido la mejor compañera (de prácticas, de gimnasio, de cañas y de todo lo demás) que podía imaginar y por todo lo que nos queda por hacer juntas. Gracias Vero, por estar siempre ahí, por ser tan cabezota como yo, por empujarme para que me aplique mis consejos. Gracias a todos por haberme dado tanto.

También tengo un huequecito para los "nuevos", que en muy poco tiempo se convirtieron en mucho. Ana con su mezcla perfecta de "buenismo" y genio, Davor porque el mundo necesita mas graciositos como tú, Diego porque siempre nos quedarán las tardes de VIPS, cines y sobre todo, Petunia y Dudley, Héctor por enseñarme a ser paciente y estar tranquila en cualquier situación, Maikel por acompañarme en tantos descansos con lo que ello conlleva, Patri, por aportar el toque extra de brillo en cada quedada. Muchas gracias a todos (y a los que me dejo) por acogerme y hacerme sentir tan arropada. Y por supuesto, a todas aquellas personas que durante estos años me han apoyado y que me han ayudado a seguir cuando las cosas se ponían muy cuesta arriba.

Para terminar, millones de gracias al verdadero pilar de mi vida: mi familia. Si hay algo que

me ha traído al punto en el que estoy ahora (y que me empujará mucho más lejos), ha sido el apoyo y el cariño de todo ese enjambre de primos, tíos y abuelos, que en mayor o menor medida, han ayudado a formar la persona que soy ahora. Gracias a Thor por enseñarme tantas cosas sin ni siquiera darse cuenta, por hacerme ver el mundo de otra manera y por recibirme siempre como si no existiera nada mejor que llegar a casa. Gracias Patri, por esas noches hablando hasta las tantas, por esos mensajes random que iluminan cualquier momento, por ser la mejor hermana pequeña que podría desear. Y gracias a mis padres, por todas las oportunidades que me han dado, por todos los sacrificios que han hecho, porque me han enseñado y empujado, me han apoyado y me han querido sin condiciones y han sabido darme ánimos cuando he estado a punto de tirar la toalla. Gracias, gracias, gracias.

Mónica Lozano Cruz
Febrero 2012

Ad astra per aspera.

A Juan Luis, Mar y Patricia.

Contents

List of Figures

List of Tables

Acronyms

TM *Template Matching*

MS *Mean Shift*

MSA *Mean Shift Adaptative*

CBWH *Corrected Background Weighted Histogram*

SOAMST *Scale and Orientation Adaptative Mean Shift Tracking*

PF *Particle Filter*

SFDA *Sequence Frame Detection Accuracy*

STDA *Sequence Track Detection Accuracy*

ATA *Average Tracking Accuracy*

ATE *Average Tracking Error*

Chapter 1

Introduction

This chapter gives an introduction to the work presented in this document. In the next sections, we describe the motivation behind this work (section 1.1), its main objectives (section 1.2) and the document structure (section 1.3).

1.1 Motivation

Computer vision is a field that pursues the automatic processing of images (for example, taken by a single camera or a set of them) to understand its content. It tries to mimic the human vision system where the brain processes images captured by the eyes [1]. The image data can take many forms, such as video sequences, views from multiple cameras, or multi-dimensional data from a medical scanner. This acquired information is later used to solve tasks or understand what happens in the scene represented by the images. This field has several applications in the areas of industrial machine vision (e.g., inspection of mechanical parts), event detection (e.g., abandoned luggage detection), forensic and biometrics (e.g., automatic face recognition)

Video tracking is an important step in many applications related with Computer Vision. It consists on locating an object or objects of interest[1] as they move in time throughout a scene by means of a vision device such as a camera [2]. A wide range of applications comes from video object tracking such as human-computer interaction, security and surveillance, video communication and compression, augmented reality, traffic control, medical imaging and video editing: some examples are depicted in Figure 1.1. Since a lot of data has to be taken into account, video tracking is considered a time consuming process, with a complexity that can be increased by the fact that object recognition techniques may have to be used.

The design of a video tracking algorithm (tracker) is a complex task. It is commonly agreed that there are three steps for its design [2]:

[1] In this document, we will use the term *target* to represent the object of interest to be tracked in the video sequence.

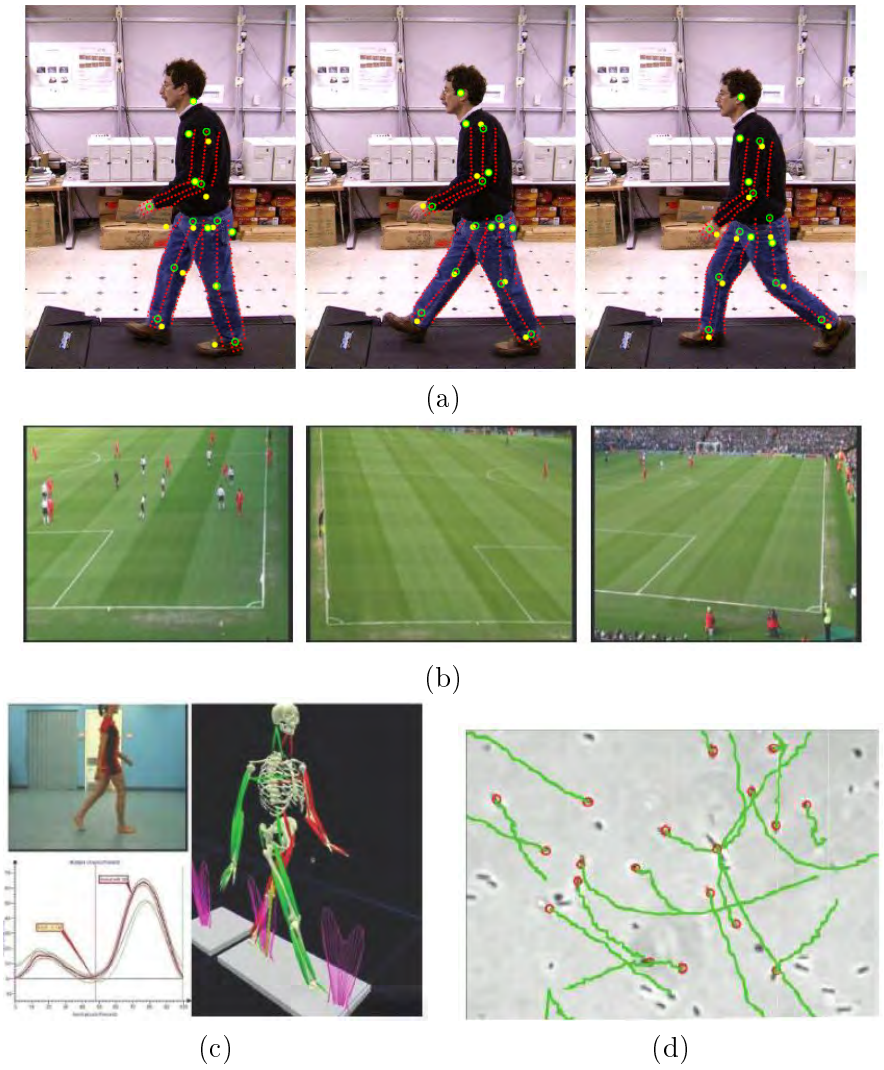


Figure 1.1: Examples of video tracking: (a) motion capture analysis [3], (b) sports tracking with multiple cameras (from PETS2003 dataset), (c) gait analysis (Oxfords Metrics Group) and (d) position tracking of *Escherichia coli* bacteria (from [4])

1) Extraction of relevant information: identify and extract the most relevant features of the target that will later be used for tracking. The better this selection is performed, the more robust the tracker will be.

2) Representation of the target: define the model for the relevant information extracted by the tracker. The ideal method of representation allows to undoubtedly identify the object while being flexible for dealing with changes such as scale, orientation, illumination, etc.

3) Propagation of the target model: use information from previously generated tracking data to estimate target parameters over time (e.g., location).

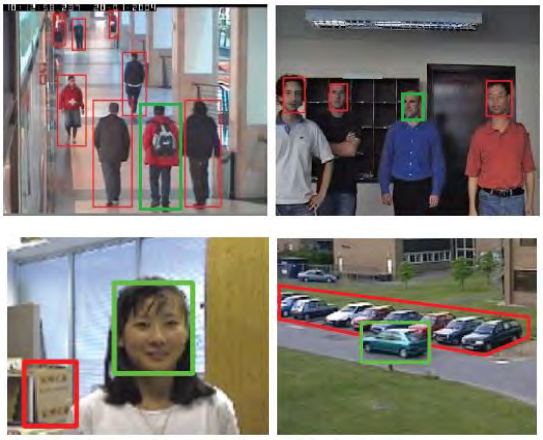


Figure 1.2: Examples of video tracking scenarios with similar objects. Target and similar objects are represented as, respectively, green and red squares

A wide variety of techniques has been developed following the above-mentioned steps. In this context, the selection of the most adequate algorithm for each application is an extremely complicated task and it is usually performed by the application designer based on his or her experience. Moreover, the high variability and complexity of the data to analyze has to be taken into account in this selection. Several issues affect the performance of the tracking algorithm such as noise, clutter[2], occlusions and changes of appearance or illumination. Hence, there is not a unique algorithm that performs perfectly in all situations. Figure 1.2 shows examples of the similar objects problem.

To precisely identify which algorithms operate better in certain situations or applications, performance evaluation has been proposed in the literature as a way to determine their strengths and weaknesses. It consists on the evaluation through the analysis of the obtained results. For performing such analysis, two main aspects have to be specified: the dataset (a set of sequences covering the situations that the tracking algorithm might face being large enough to represent the real world conditions) and the metrics to measure the precision of tracking algorithms (which allow to quantify how well the algorithms perform). These two aspects are also known as the evaluation protocol of video tracking [2]. Traditional approaches use metrics based on ground-truth information that represents a manual annotation of the ideal tracking result and it is manually annotated. The generation of the ground truth is usually a time consuming step and, therefore, limits the amount of data in the dataset. Although there are other approaches not focused on ground-truth information [5], most of the current literature assumes the availability of such data. Furthermore, the existence of several metrics increases the complexity of designing an evaluation protocol. Another point to be taken into account is the increasing quantity of video

[2]In this document, we use *clutter* to represent when the extracted features of other objects and of the background are similar to the ones of the target being tracked (e.g., target appearance).

data available, which generates a new need to automate the whole tracking evaluation process.

Current comparative studies of tracking algorithms are limited due to the low number of employed data and the similarity of the analyzed metrics. For example, [6] proposed some metrics that provide redundant information. Hence, it is not easy to extrapolate the achieved conclusions to the analysis of new sequences. Another problem is that there are not studies comparing algorithms of diverse nature with different situations that may affect tracking performance. This fact also limit the conclusions of the performed evaluations.

As explained earlier, the available datasets and benchmarks have proven not be enough to meet the needs for the current video tracking applications [7]. Currently, there is no standard evaluation procedure that allows an easy comparison between algorithms, showing both strengths and weaknesses of each one. Since each proposed evaluation protocol uses different datasets and metrics, it is not possible to provide general and standard evaluation scores which would allow to compare different algorithms in each scenario.

1.2 Objectives

The main objective of the work presented in this document is to develop an evaluation protocol for estimating the performance of video object tracking algorithms. This protocol has to consider the main issues that affect the performance of video tracking algorithms. In this work, we focus on the evaluation of single-object tracking. In order to do so, the following points are addressed:

- In-depth study of the state of the art. It includes the review of the tracking-related work considering the required analysis stages, the existing approaches, the main tracking issues, the evaluation metrics and the available datasets.

- Selection and implementation of the most representative tracking approaches. Among existing literature, the most popular approaches for deterministic and probabilistic tracking are selected and implemented for their evaluation.

- Creation of an appropriate dataset for the evaluation of video tracking. It consists on the design of synthetic and real sequences that represent the most important tracking issues. Moreover, the use of existing real data has been considered to compose the dataset.

- Evaluation of existing metrics for video object tracking. The objective is to study the relation between all the existing metrics for tracking evaluation. Thus, it will allow to determine the best metric to use for the evaluation task.

- Design and implementation of a protocol for evaluating the performance of video tracking. It proposes an evaluation methodology covering different types of problematic tracking situations.

- Application of the proposed evaluation protocol. The performance of the selected tracking algorithms is tested by using the previously-defined protocol.

1.3 Document structure

The structure of the document is as follows:

- Chapter 1. This chapter presents the motivation and the objectives of this work.
- Chapter 2. This chapter discusses the literature related to the presented work.
- Chapter 3. This chapter describes the selected tracking approaches that will be evaluated.
- Chapter 4. This chapter overviews the proposed protocol for evaluating video object tracking. It includes the aspects considered and the dataset created.
- Chapter 5. This chapter provides the experimental result by giving an in-depth analysis of the application of the evaluation protocol to the selected algorithms.
- Chapter 6. This chapter summarizes the main achievements of the work, discusses the obtained results and gives suggestions for future work.

At the end, several appendices list further details:

- Appendix A. This appendix provides a brief description of performance evaluation approaches for multiple object tracking.
- Appendix B. This appendix describes the main existing datasets related with video tracking.
- Appendix C. This appendix describes in detail the sequences that comprise the created dataset in order to test the evaluation protocol proposed.
- Appendix E. This appendix expands the study of how different errors in the initialization affect the performance of different algorithms in sequences with selected issues.
- Appendix F. This appendix contains the Introduction chapter in Spanish.
- Appendix G. This appendix contains the Conclusions and Future Work chapter in Spanish.

Chapter 2

Related Work on Video Tracking

This chapter gives an overview of existing literature in the scope of the work presented in this document. In the next sections, we describe the main aspects of video tracking (section 2.2), the metrics for performance evaluation (section 2.3) and the datasets (section 2.4).

2.1 Introduction

Video object tracking studies the problem of estimating the trajectory of an object of interest (target) in an image sequence. Tracking is a complicated task due to high variability and complexity of the data to analyze. When designing a tracking algorithm, the following steps have to be taken into account such as feature extraction, target representation (i.e., object modeling) and propagation of the target model over time. In order to solve the above-mentioned problems, a large variety of techniques has been developed. In the first section, we review these three steps and categorize the existing approaches.

In this context, selecting the most appropriate technique in each situation is an extremely complicated task, usually performed by a human operator based on their experience. Several issues have to be considered: noise in images, complex object movement, similar objects in scene (clutter), partial or total occlusion, changes in the illumination and computational complexity (critical for real-time applications). As a solution to this problem, current literature proposes the evaluation of tracking algorithms using the obtained results. In order to do so, different metrics have been defined, capable of evaluating the accuracy of the algorithm, and simple data sets have been designed. However, comparative studies are currently not enough due to the low number of data used as well as the similarity of the analyzed techniques. This limits the conclusions that can be drawn from the evaluation process. One of the critical challenges is to correctly match (spatially and temporally) the algorithms output's and ground-truth annotations. Therefore, several initiatives have been proposed for providing such metrics: VACE Metrics [8], CLEAR Metrics[9] and PETS Metrics [10, 6] among others. In the second section, we discuss the literature

related to performance evaluation of video tracking.

As mentioned in [11], ideally, the field of visual tracking would contain a dataset of standard videos that would be universally used by researchers and each dataset would have its particular quantitative metrics to define the success or failure of an algorithm. Some initiatives have been proposed in order to develop such dataset which are mentioned in the last section of this chapter and described in Appendix B.

2.2 Description

Recently, several surveys have been published for video object tracking [12, 11, 2]. In this section, we synthesize their most relevant information considering the stages involved, the features, the representation, the prediction of motion and the tracking algorithms.

2.2.1 Stages involved

Tracking-based video analysis systems are typically composed by (at least) three modules as shown in Figure 2.1. They are:

- Target detection: it recognizes the target to be tracked. Some of the methods used are: point detectors [13], segmentation [14], foreground detection [15] and supervised learning [16].

- Tracking: it performs the tracking analysis with the aim of locating the target in each frame[1]. The chosen representation and motion model selected for each object completely determines the movement and deformation it may have.

- Analysis of tracking data: it studies all the tracking data generated to fulfill the purposes of the application (e.g., trajectory creation, outlier detection, event recognition).

Figure 2.1: Stages in a typical tracking-based video analysis system.

[1] In this document, we will use the term frame and image interchangeably

2.2.2 Features

All tracking systems require to define the features to used for representing the target[2]. It is fundamental that the chosen features uniquely identify the target from the background of the scene or other objects, therefore no feature is said to be better than the others, since it depends on the situation where the tracking algorithm will be used. As an example, if a soccer player movement is to be followed, color is not the best feature as other players share this characteristic. Moreover, the selected feature has to be descriptive as well as flexible for dealing with changes such as rotations, illumination and scale changes. In most of the existing approaches, the user manually chooses these features, depending on his/her experience and the final application. These features can be grouped in three areas: low-level, mid-level and high-level [2].

2.2.2.1 Low-level features

Color The color of a region is determined by finding the mean value for all the pixels in a certain region [12]. The problem with this representation is that when dealing with multimodal color distributions the mean value is not enough. For example, when tracking a red suitcase the mean value would be sufficient. However, if a person with this particular suitcase has to be tracked, the representation fails. To solve this issue, color histograms (where the number of times a color appears is counted) were introduced. These histograms are widely extended due to their simplicity and good performance. One of their advantages is that changes in the image such as rotation and translation do not affect them, and if the target rotates, is occluded or changes its scale, the histogram does not suffer a significative change. On the other hand, since histograms collapse all the information it is impossible to know which color comes from which part of the image, therefore, losing the spatial structure of the object color. Also note that illumination changes have a great effect in color histograms. To mitigate this problem, color spaces such as HSV (hue, saturation, color) are used, since they are more robust than RGB spaces. However, since each color space has its advantages and disadvantages, it is not possible to pick one as the optimum for all scenarios .

Gradient and derivatives Histograms of oriented gradients are an alternative to the color histograms mentioned above, and they show the orientation distribution of gradient vectors within the region [2]. They are more robust to illumination changes than color ones as edges tend to persist with such changes. However, background clutter negatively affects the extraction of the target gradient. In Figure 2.2 an example of gradient edge detection is depicted.

[2]It is important to note that the object's feature set is not to be confused with its template. For example, a feature could be the intensity of the pixels while the template (or model) would be the specific numerical values those intensities can take

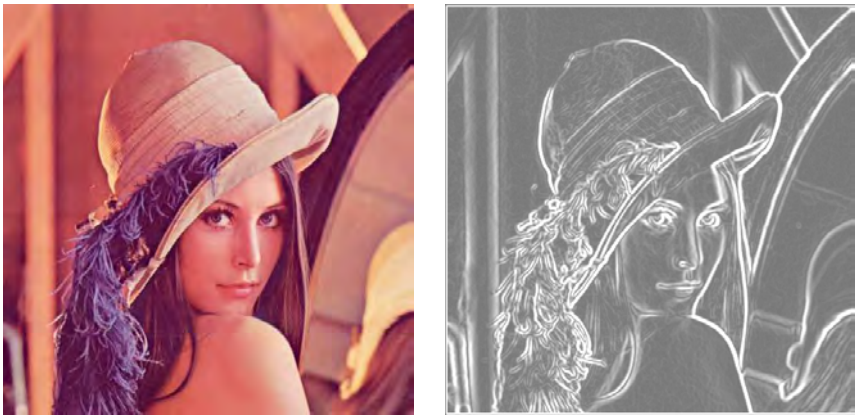


Figure 2.2: Gradient edge detection example

2.2.2.2 Mid-level features

Edges and lines Edges and lines have been used widely through the years, and they became one of the most used methods [17]. An edge is a pixel located in the boundary between two different regions (each with its constant intensity).

Interest points An interest point is a pixel in which its local image structure (i.e., a particular neighborhood of this point) contains meaningful information for image or video analysis (e.g., the corner of a table, the body's joints). An example of features used in tracking are the Scale Invariant Feature Transform (SIFT) [18] and the Speeded Up Robust Feature (SURF) [19].

2.2.2.3 High-level features

Instead of grouping mid-level features, other option to define a target is to detect it as a whole based on its appearance. There are two different approaches in this case, depending on which part is modeled (background or foreground) [2]. The former determines which part of the image comprises the fixed model and the rest of objects that can't be explained by means of that model are labeled as possible targets. Then, these objects are tracking using combinations of low-level features [20]. In the later, the appearance of a pre-defined class of targets is obtained by learning representative features of the selected class. An example is color-based segmentation, used to detect faces of people. Then, detected faces are similarly tracked as described above [15].

2.2.3 Representation

For providing an accurate video tracking, the features of the target have to be properly represented by means of a model. For this representation, there are several approaches: basic (where the target is represented as a single point), patch or volume approximations (where the features

are extracted from the selected area as, for example, color histograms and intensity templates), articulated (where a combination of rigid models approximate the shape of the target), and deformable (where fluid models, contours or point distribution models are used). Some examples of shape representation can be found in Figure 2.3.For example, this target model may contain information regarding its shape and appearance [2].

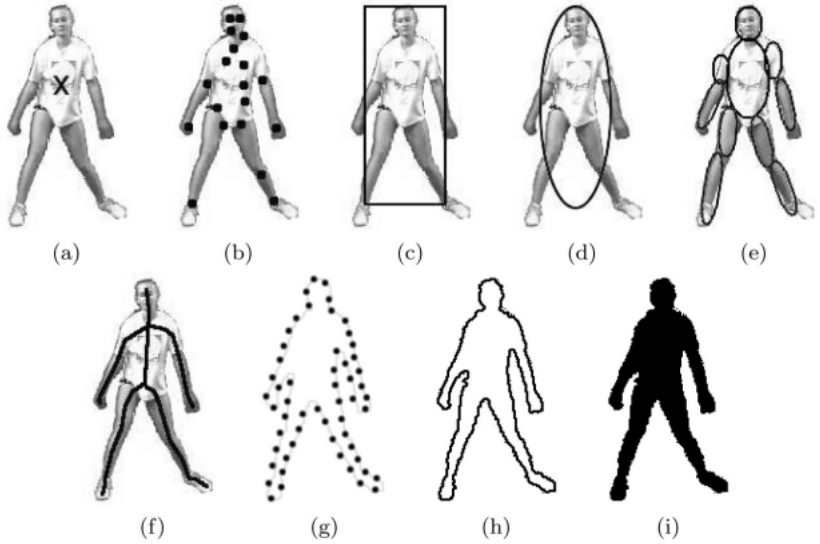


Figure 2.3: Representation: a) centroid, b) multiple points, c) rectangular patch, d) elliptical patch, e) part-based multiple patches, f) object skeleton, g) complete object contour, h) control points on object contour, i) object silhouette. Extracted from ([12])

2.2.4 Prediction of motion

The prediction of motion of a target (also known as motion estimation) determines the movement of the target for performing an efficient video tracking analysis. It can be considered as a way to reduce the computational burden since guessing where the target is going to be in each frame helps to speed up the whole tracking process. Motion estimation tries to explain how the target moves by means of a correspondence vector (describing the displacement of a pixel between two frames) or by an optical flow field (which contains information regarding the velocity if apparent motion exists).

For locating the target in a certain region of the image, we distinguish two approaches is an exhaustive (deterministic) or selective (probabilistic) search is performed on the search area.

2.2.4.1 Deterministic approach (exhaustive search)

This approach could be view as an optimization problem without any knowledge about the motion model of the target. The goal is to identify and minimize the cost function that defines

(a) (b)

Figure 2.4: Prediction of motion approaches: (a) Exhaustive search, (b) Selective search (where blue points represent the selection)

the similarity between the target to be tracked and the features observed in the search area of the current frame. Hence, an exhaustive search is performed for locating the target. The most elemental way to use this information is to begin the analysis in the same position as the target occupied in the previous frame. If the frame rate of the sequence is high it is obvious the advantage of beginning the analysis in the same position previously determined, since it decreases significantly the search time. As seen in Figure 2.4(a), the blue box represents the search area. It is located in the position of the target in the previous frame (the target is depicted by means of the smaller box). If the object does not change its position abruptly, this method provides good results and optimizes the search time. This example corresponds to the application of the MeanShift algorithm [21].

2.2.4.2 Probabilistic approach (selective search)

This approach assumes a certain motion model of the target and consists on two stages: prediction (considering previous data) and update (considering image data). This prediction uses the motion model and projects forward the current state of target (e.g., target location and size) from the previous frame to the current one. The update stage checks the similarities among the predictions and the target model (e.g., color histogram similarity) and does not belong to prediction of motion.

In current literature, two approaches are widely used: Kalman [22] and Particle [23] filters. Kalman filter is a tool devised to solve estimation problems which are linear and the noise is Gaussian. Particle filters allow the system (e.g., the target) to behave non linearly without any Gaussian assumption. In Figure 2.4 (b) the Particle filter approach is represented as an example of the selective search. The particles (blue points) describe each point where a new search is performed.

2.2.5 Taxonomy for tracking algorithms

A combination of different taxonomies from [12, 2] is proposed, and is depicted in Figure 2.5

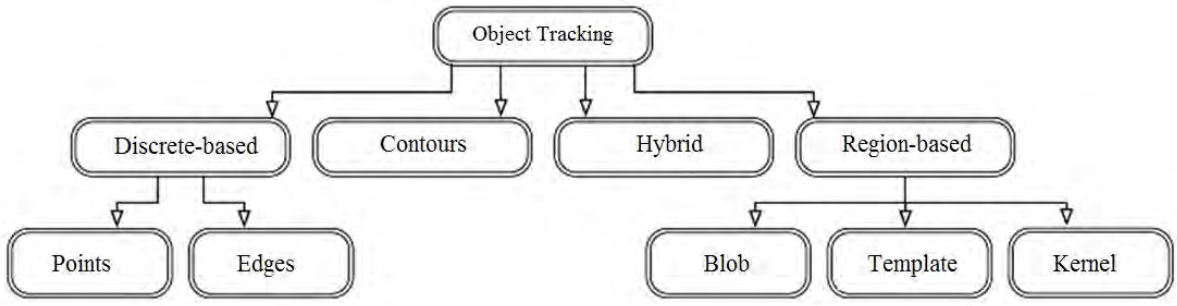


Figure 2.5: Object tracking taxonomy

2.2.5.1 Discrete-based trackers

Discrete features trackers use simple image structures such as points, lines and edges. These trackers use points to represent detected targets in consecutive frames and the association of the points is based on the previous target state which can be defined as the target location and motion[12].

Points The information regarding position and motion is included in the previous state. For target detection it is necessary an external mechanism, and then feature points are extracted to represent the target. One of the greatest advantages of point trackers its their ability to track small objects (represented by a single point). For each point, a cost of association between the point in the previous frame an the point in the current one is calculated. There are several constraints that affect this cost: proximity, velocity, smooth motion and rigidity among others. Since this method provides support for several different situations it has been widely studied and there are several works regarding deterministic point tracking [24].

Edges Two different approaches can be taken: a previous model can be used or not. If no model is used, the tracking is performed by means of one or more Kalman filters and it is repeated in each frame. If a 3D model of the object is used it is necessary to develop complex transformations to align the model with the image content. Because of its complexity, 3D models approach has experienced a significant decrease of attention since the beginning of the 1990s [25].

2.2.5.2 Region-based

Region based trackers are known for the wide variety of features they use in order to represent and track the target. Some of these features are color, texture, intensity and gradient. The trackers can be divided in three groups: blob, template, and kernel.

Blob Blob trackers use very specific information and details regarding the target followed, for example, the average color or centroid position. These trackers depend on a previous stage that detects target candidates as blogs (i.e., background substraction). Their main advantage is their high effectiveness when following a target with a stationary camera or widely separated targets and their low complexity allowing the development of real-time systems [26]. However, some complications may arise when the targets move close to the camera.

Template Template trackers describe a feature (e.g., the intensity value) of the target. There are two approaches depending if the template is rigid or deformable. Rigid template tracking [27] assumes that the target information is not going to change in consecutive frames. Thus, simple metrics can be used to locate the target (e.g., intensity correlation). For deformable template tracking [28], a parametric motion model is employed to define the movement of the target and, therefore, apply transformations to the template of the target. The main problem of this approach is the requirement of a learning stage that might not be possible due to data availability.

Kernel Kernel histogram methods basically use a weighting kernel as well as a histogram to represent the target. These trackers can be viewed as a the midpoint between blob trackers and template trackers. The available algorithms differ in several aspects such as the target tracked, the selected features and the method to model the target motion. MeanShift tracker [21] is one of the most popular kernel trackers.

2.2.5.3 Contours

A contour is a curve (open or close) that outlines a target. The main difference between contour trackers and edge trackers is that the later focuses on straight lines, while the former can follow targets with contour deformations. Also note that the approaches for these two categories are completely different, hence they belong to different categories. Contour trackers can be divided in three approaches: basic (with rough approximations of area or volume), articulated (with more complex models of the target including several rigid models) and deformable [2]. It is assumed that there are only small changes in the shape of the target and position, so the location in each frame is done by slightly changing the information from the previous frame. There have been several approaches during the years, including the use of different techniques such as basic

snakes [29] (using discrete curve parametrization), level sets [30] and trackers that used region information [31].

2.2.5.4 Hybrid

This category represents a new trend that combines the previously described approaches with the object of improving the overall tracking process. In [32] a combination of MeanShift and Particle Filter was proposed, where the tracker first produces a smaller number of samples using Particle Filter and then shifts the samples toward a close local maximum using MeanShift. This improves the accuracy (with better results than the isolated application of both tracking algorithms) while using less samples than Particle Filter alone.

2.3 Performance evaluation metrics

In this section, we describe the most representative metrics that have been proposed for performance evaluation of single-object video tracking.

2.3.1 Nomenclature

We will fist begin by introducing the required notation for describing complex metrics [6, 33].

- TP: True positive, a target pixel appears both in the ground-truth annotation and the algorithm result (per frame).
- TN: True negative, a target pixel that appears neither in the ground-truth annotation nor the algorithm result (per frame). This metric is a little confusing since it is not very clear what a true negative target is. However, it is used in some works that perform a pixel level evaluation.
- FP: False positive, a target pixel that appears in the algorithm result, but not in the ground-truth annotation (per frame).
- FN: False negative, a target pixel that appears in the ground-truth annotation but not in the algorithm result (per frame).
- $GT_i^{(t)}$: represents the i^{th} ground-truth annotation for the t^{th} frame. Note that there is no identificator of the target as we focus on single-object tracking (i.e., there is only one)
- $D_i^{(t)}$: represents the estimated location of the i^{th} target for the t^{th} frame .
- $N_{GT}^{(t)}$: represent the number of ground-truth annotations in the t^{th} frame. In our case, this number is equal to 1.

- $N_D^{(t)}$: represent the number of target annotations in the t^{th} frame. In our case, this number is equal to 1.

2.3.2 Initiatives for performance evaluation

VACE Metrics The Video Analysis and Content Extraction (VACE) initiative was developed for evaluating object detection and tracking in video sequences. Its main objective is the creation of algorithms and implementations for automatic video content extraction, multimodal fusion and event understanding [8]. In both Phases I and II an in-depth study of video content analysis was made, being object detection and tracking the primary focus on VACE-II. Several breakthroughs were made regarding automated detection and tracking of scene objects (faces, hand, bodies, vehicles). VACE Metrics provide a measurement of the whole system performance including cases with missing objects, false positives, etc.

CLEAR Metrics The Classification of Events, Activities and Relationships (CLEAR) evaluation and workshop was the first international effort that evaluated different systems designed to recognize events, activities and their relationships [34]. Its main objective was to get together different researchers so a common international evaluation could be established, combining two programs: VACE and CHIL[35]. A need of a standardized and unified set of metrics (valid for both programs) appeared to get both initiatives together. The results of the workshop generated great advantages, including more data available for the research community as well as the evolution of some widely accepted performance metrics [8].

PETS The first Performance Evaluation of Tracking and Surveillance (PETS) took place in 2000. It was created due to the growing necessity to develop a systematic performance evaluation for video tracking techniques. The PETS workshop was created to propose different datasets addressing common problem of video tracking. It was the beginning of the performance evaluation activities (which later included ETISEO, for example). Each year since 2000 a PETS workshop has taken place (in different countries) proposing a different challenge for researchers.

ETISEO ETISEO (Evaluation du Traitement et de l'Interpretation de Séquences Video) is a Video Understanding Evaluation project which ended successfully in December 2006 [36]. The project took place for two years and was part of the Techno-Vision evaluation network funded by the French ministry of defense and the French ministry of research. The main goal of ETISEO was to evaluate vision techniques for video surveillance, focusing on the treatment and interpretation of videos involving pedestrians and/or vehicles [37]. The four main project objectives are: acquisition of precise knowledge of vision algorithms, productive discussions between research labs and companies, creation of two ontologies to facilitate the communication

between all participants in the domain and development of automatic evaluation tools for vision algorithms.

2.3.3 Single target evaluation

For evaluating the tracking of single targets, several approaches exist. For example, [10] proposed a classification of the metrics based on the approach taken: frame-based and object-based metrics. Frame-based metrics test each frame individually and then the result is averaged over the whole sequence. Object-based metrics perform the evaluation over the whole trajectory of each object. Following this categorization, we describe the related approaches.

2.3.3.1 Frame-based metrics

The following measures are the most common used, and they use the previously described nomenclature:

- Precision [38]: indicates the rate of false positives at pixel level for each frame.

$$Prec = \frac{\#TP}{\#TP + \#FP}. \tag{2.1}$$

- Sensitivity [38]: indicates the rate of false negatives at pixel level for each frame . Also called True Positive Rate (TPR).

$$Sens = \frac{\#TP}{\#TP + \#FN}. \tag{2.2}$$

- F-Measure [38]: this measure provides a relation between Precision and Sensitivity and allows to emphasize one or the other (depending on the situation) by means of α.

$$F - Measure = \frac{1}{\alpha \cdot (\frac{1}{Sens}) + (1-\alpha) \cdot (\frac{1}{Prec})} = \frac{\#TP}{\#TP + \alpha \cdot \#FN + (1-\alpha) \cdot \#FP}. \tag{2.3}$$

- F-Score [38]: if both measures (Precision and Sensitivity) are equally important, $\alpha = 0.5$ and the metric is called F-Score.

$$F - Score = \frac{\#TP}{\#TP + \frac{1}{2} \cdot (\#FN + \#FP)}. \tag{2.4}$$

Also, based on the same basic notions, more complex formulas are introduced:

- Tracker Detection Rate [10]:

$$TRDR = \frac{\#TP}{\#GT}. \tag{2.5}$$

- False Alarm Rate [10]:

$$FAR = \frac{\#FP}{\#TP + \#FP}. \tag{2.6}$$

- Detection Rate [10] (equal to sensitivity):

$$Detection\ Rate = \frac{\#TP}{\#TP + \#FN}. \tag{2.7}$$

- Specificity [10]: indicates the rate of false positives in relation to the total number of negatives. Also called True Negative Rate (TNR). As already mentioned, true negative only applies to pixels or frames, so this metric should only be used for those elements.

$$Spec = \frac{\#TN}{\#TN + \#FP}. \tag{2.8}$$

- Accuracy [10]:

$$Accuracy = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN}. \tag{2.9}$$

- Positive Prediction [10] (equal to precision):

$$Positive\ Prediction = \frac{\#TP}{\#TP + \#FP}. \tag{2.10}$$

- Negative Prediction [10]:

$$Negative\ Prediction = \frac{\#TN}{\#FN + \#TN}. \tag{2.11}$$

- FN Rate [10]: the rate of positives that were previously assigned as negatives.

$$FNR = \frac{\#FN}{\#FN + \#TP} = 1 - Sens. \tag{2.12}$$

- FP Rate [10]: the rate of negatives that were previously assigned as positives (only defined for pixels and frames).

$$FPR = \frac{\#FP}{\#FP + \#TN} = 1 - Spec. \tag{2.13}$$

FDA The Frame Detection Accuracy measure [39] calculates the spatial overlap between the ground-truth annotation and the estimated target location as a ratio of the spatial intersection between them and the spatial union of them for a given frame.

$$FDA(t) = \frac{OverlapRatio}{\frac{N_{GT}^{(t)} + N_D^{(t)}}{2}}, \tag{2.14}$$

and

$$OverlapRatio = \sum_{i=1}^{N_{mapped}^{(t)}} \frac{|GT_i^{(t)} \cap D_i^{(t)}|}{|GT_i^{(t)} \cup D_i^{(t)}|}, \tag{2.15}$$

where N_{mapped} is the number of matched pairs of ground-truth annotation and estimated target location. For single-object tracking $N_{mapped} = 0$ if there is no matching (or $N_{mapped} = 1$ if there is).

Normalized spatial overlap This measure determines the amount of overlap between the ground-truth annotations and estimated target locations. It is computed in every frame where the target exists [7].

$$O = \frac{|TP|}{|TP| + |FP| + |FN|}, \tag{2.16}$$

where k denotes the frame. The higher the overlap, the better the tracking. As it can be observed, this measure is similar to FDA.

Displacement error rate This measure determines the displacement between centroids for ground-truth annotation and detection target locations [40].

$$DER = \frac{displacement\ error\ between\ ground-truth\ and\ estimated\ position}{size\ of\ the\ target} \tag{2.17}$$

This measure presents an issue since the size of the ground-truth annotation is not taken into account, and therefore does not provide accurate results when it changes.

2.3.3.2 Object-based metrics

SFDA The Sequence Frame Detection Accuracy measure calculates in each frame the spatial overlap between the estimated target location and the ground-truth annotation. This mapping is optimized on a frame-by-frame basis. It contains information regarding the number of objects detected, missed detects, false positives and spatial overlap, providing a ratio of the spatial intersection and union between two object locations. The total sum of data from the FDA is then normalized to the number of frames including ground-truth targets. Therefore, SFDA [39] can be seen as the average of the FDA over all the relevant frames in the sequence. We obtain

the SFDA formula, which ranges from 0 to 1; the higher the value, the better.

$$SFDA = \frac{\sum_{t=1}^{t=Nframes} FDA(t)}{\sum_{t=1}^{t=Nframes} \exists(N_{GT}^{(t)} \, OR \, N_{D}^{(t)})} \tag{2.18}$$

As seen in Figure 2.6, when $SFDA = 1$ (first case), the ground-truth annotation and the estimated target locations are perfectly aligned. In cases where $SFDA \neq 1$ (second image) a difference in the aligning is visible.

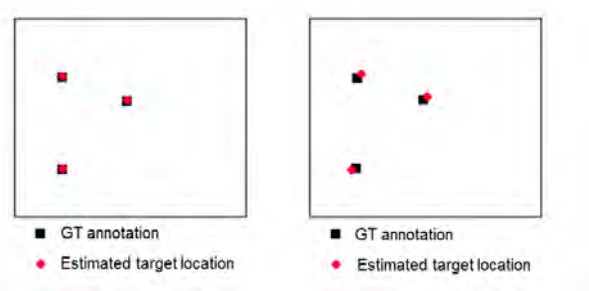


Figure 2.6: Example for SFDA metric.

ATA (Average Tracking Accuracy) The Average Tracking Accuracy measure contains information regarding the number of objects detected and tracked, missed objects and false positives. It is defined as the average ratio of the spatial intersection and union of the ground-truth object and the tracked object over all the frames. The mapping was optimized at a sequence level and it penalizes fragmentation in both the temporal and spatial dimensions. The Sequence Track Detection Accuracy was calculated by establishing a one-to-one mapping (computing the measure of all ground-truth and detected object combinations) and maximizing the score for the sequence. The main difference is that while in tracking the overlap is computed in the spatio-temporal dimension, in detection only the overlap in the spatial dimension is studied.

The STDA [39] formula also varies from 0 to 1; the higher the value, the better.

$$STDA = \sum_{i=1}^{N_{mapped}} \frac{\sum_{t=1}^{Nframes} \frac{|GT_i^{(t)} \cap D_i^{(t)}|}{|GT_i^{(t)} \cup D_i^{(t)}|}}{N_{(GT_i \cup D_i \neq 0)}}. \tag{2.19}$$

Observe that STDA corresponds to the SFDA when there is a matching between ground-truth annotation and the estimated target location ($N_{(GT_i \cup D_i \neq 0)}$). The average over all the objects in the sequence can be obtained [39]:

$$ATA = \frac{STDA}{\frac{N_{GT}^{(t)} + N_D^{(t)}}{2}}. \tag{2.20}$$

The numerator in the expression rewards true positives and penalizes false alarms. This computes the percentage of ground-truth that is covered with mapped objects [41].

However, the definition of ATA is not unique, and another expression can be found in [42]:

$$ATA = \frac{1}{N_{frames}} \sum_{t=1}^{N_{frames}} \frac{\left| GT_i^{(t)} \cap D_i^{(t)} \right|}{\left| GT_i^{(t)} \cup D_i^{(t)} \right|} \tag{2.21}$$

Both SFDA and ATA metrics provide a way to condense the tracking data in a single score, thus allowing to determine a trend of tracking performance. The main drawback of this metrics is the inability to provide an identification of failure components [43].

ATE (Average Tracking Error) The Average Tracking Error proposed in [42] can be seen as a false positive rate (whereas ATA represents the true positive rate). It provides a ROC-like curve which allows to compare and evaluate the tracker's performance.

$$ATE = \frac{1}{N_{frames}} \sum_{i=1}^{N_{mapped}} \frac{\left| D^t \setminus GT^t \right|}{|D^t|} \tag{2.22}$$

where $\left| D^t \setminus GT^t \right|$is the relative complement, that is, the set of elements in B, but not in A.

OTE (Object Tracking Error) The Object Tracking Error proposed in [10] aims to calculate the average discrepancy between the ground-truth bounding box and the system result centroids.

$$OTE = \frac{1}{N_{frames}} \sum_{i \in g(t_i) \bigwedge r(t_i)} \sqrt{\left(x_i^g - x_i^r\right)^2 + \left(y_i^g - y_i^r\right)^2}, \tag{2.23}$$

where x_i^g and y_i^g are the x-coordinate and y-coordinate of the centroid of the target in i^{th} frame of ground-truth and $x_i^g$and x_i^g are the x-coordinate and y-coordinate of the centroid of the target in i^{th} frame of the tracking result. This metric does not work properly in some cases due to the fact that different areas can have the same centroid and therefore provide the same OTE metric even though the discrepancy is not the same.

Closeness of Track This metric proposed in [6]aims to calculate the average closeness between a pair of ground-truth and system results tracks.

$$CT\left(GT_i, D_i\right) = \left\{ A\left(GT_i^{(1)}, D^{(1)}{}_i\right), ... A\left(GT_i^{(t)}, D^{(t)}{}_i\right) \right\} \tag{2.24}$$

, where A represents the spatial overlap for ground-truth and system tracks.

The closeness of the whole sequence can be averaged by weighting the CT of all pairs.

$$CTM = \frac{\sum_{t=1}^{M} CT_t}{\sum_{t=1}^{M} length\left(CT_t\right)} \tag{2.25}$$

, where CT_t is the Closeness of Track for the t^{th}pair of ground-truth annotation and estimated location, and $length(CT_t)$ can be viewed as the total number of frames since A will always have a value (0 if there is no spatial overlap)

The weighted standard deviation of track closeness can be obtained for the whole sequence:

$$CTD = \frac{\sum_{t=1}^{M} length\left(CT_t\right) \times std\left(CT_t\right)}{\sum_{t=1}^{M} length\left(CT_t\right)} \tag{2.26}$$

Track Completeness This metric proposed in [6] is defined as the time span that there is overlap between the system and ground-truth tracks and divided by the duration of the ground-truth track.

$$TC = \frac{\sum_{t=1}^{N_D^{(t)}} O\left(GT^{(t)}, D^{(t)}\right)}{N_{GT}^{(t)}}, \tag{2.27}$$

where $O\left(GT^{(t)}, D^{(t)}\right)$ is a binary variable with value 1 if a pair is overlapped more than a threshold value and 0 otherwise.

The average track completeness for the whole sequence can be described as:

$$TCM = \frac{\sum_{i=1}^{l} max\left(TC_i\right)}{N_{GT}^{(t)}}, \tag{2.28}$$

, where TC_i is the Track Completeness in frame i.

Lost Track Ratio In [7] a new metric to calculate the loss of targets is proposed. A target is said to be lost when the spatial overlap (Equation 2.16) between the ground-truth and the estimated target is smaller than a threshold. Afterward, the lost-track ratio (λ) is calculated based on the overlap of the sequence:

$$\lambda = \frac{N_{frames}}{N_{(GT_i \cap D_i = 0)}}. \tag{2.29}$$

Because the appropriate value of τ is different for different tracking applications, it was considered in [7] the variation of τ for a full range of values, from $\tau = 0$ to $\tau = 1$ with an increment of 0:01. We refer to these parametrized values of the lost-track ratio as $(\lambda(\tau))$. In order to standardize this measure so that it can be used independently of the tracking application,

the Area Under the lost track ratio Curve is computed.

$$AUC_\lambda = \sum_{\tau=0}^{1} \lambda(\tau) \tag{2.30}$$

The lower this value is, the better. Therefore, a tracking algorithm with perfect performance would provide a $AUC_\lambda = 0$, whereas an algorithm that never finds the target (in all frames of the sequence) would deliver and $AUC_\lambda = 1$. An example of two curves depicting two scenarios is presented in Figure 2.7. Following the analysis, the tracking result of the left, with a $AUC_\lambda = 0.516$ is better than the one on the right with a $AUC_\lambda = 0.847$ due to a lower value under the curve of the lost-track ratio.

In following chapters, AUCinv will be used defined as the inverse of the AUC_λ:

$$AUCinv_t = 1 - AUC_t \tag{2.31}$$

where t is the frame number ranging from 0 (1) to indicate worst (best) performance.

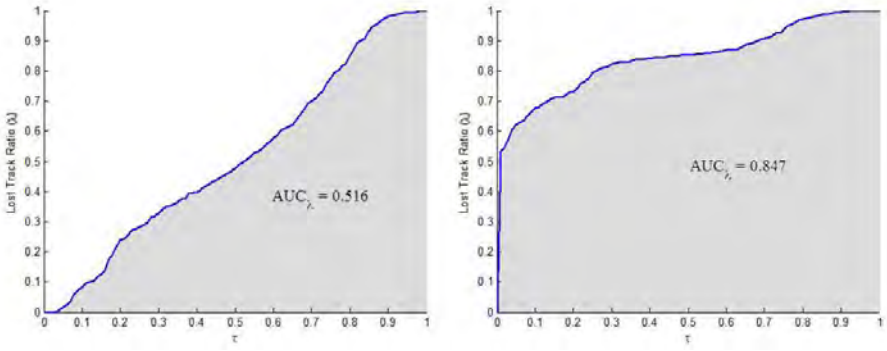


Figure 2.7: Lost Track Ratio examples (from [7])

2.3.4 Summary

Once all metrics were studied, a selection of the most interesting ones was done and can be found in Table 2.1. These metrics were selected due to simplicity and accuracy, and also as they seem to provide different information regarding tracking results. Single target metrics depicted in Table 2.1 will be used in the study of the evaluation protocol.

Target	Classification	Metrics
Single target	Frame-based	SFDA
	Object-based	ATA, ATE, AUCInv, TC

Table 2.1: Selected metrics for performance evaluation of video object tracking

2.4 Datasets

Several datasets were analyzed (in order to obtain a wide view of available data) and are detailed in Appendix B. A summary of the information regarding those datasets is summarized in Table **??**, which shows that there is a lack of datasets with enough test sequences (and corresponding annotations) to analyze some of the most important issues in video tracking. The Table includes information obtained by a thorough study of each dataset so an understanding of the issues presented in each sequences was obtained.

Dataset		Description	# Seq	GT	AvailableDesc	ComplexMov	Illumination	Occlusion	SimilarObjs	ScaleCha
SPEVI	Single	Single person/face detection and tracking	5	Yes	Sequence	√	√			√
	Multiple	Multiple person/face detection and tracking	3	Yes	Sequence		√	√	√	
ETISEO		Indoor and outdoor scenes	86	Yes	No	√	√	√	√	√
PETS	2000	Outdoor people and vehicle tracking	1+1	No	No		√		√	√
	2001	Outdoor people and vehicle tracking	5+5	No	No	√	√	√		
	2006	Person and baggage detection in train station	28	No	No		√	√	√	√
	2007	Loitering, attended and unattended luggage	1+9	No	No	√	√	√	√	√
	2010	Crowd activities	1+3	No	No	√	√	√	√	√
CAVIAR		People tracking in a mall	17	Yes	No		√	√	√	√
VISOR	Indoor	Indoor people tracking with occlusions	6	No	No	√		√	√	√
i-Lids		Abandoned luggage, parked vehicle	7	No	No	√	√	√	√	√
Clemson		Head tracking	16	Yes	Sequence	√	√	√	√	√
MIT Traffic Dataset		Vehicle tracking	20	No	No	√	√	√	√	√

Table 2.2: Study of the characteristics of detailed datasets.

Chapter 3

Selected Tracking Algorithms

This chapter presents the selected algorithms to be evaluated in this project. First, a brief introduction including an overview of the algorithms is presented (section 3.1). In the next sections, an in-depth description of the following deterministic tracking algorithms is explained: Template Matching (section 3.2.1), MeanShift (section 3.2.2), Adaptive MeanShift (section 3.2.2.5), Corrected Background Weighted Histogram (section 3.2.3) and Scale and Orientation Adaptive MeanShift Tracking (section 3.2.4). The the probabilistic Particle Filter framework is detailed (section 3.3.1), followed by a presentation of the algorithm for color-based particle filters (section 3.3.2).

3.1 Introduction

As mentioned in Section 2.2.5, kernel-based video tracking is one of the main categories in which region trackers are grouped. Kernel histogram methods basically use a weighting kernel as well as a histogram to represent the target. These tracking algorithms can be viewed as the midpoint between blob and pixel oriented algorithms. Hence, we have selected the most relevant deterministic and probabilistic kernel-based algorithms.

The most basic tracking algorithm is Template Matching (TM), in which a model of the target is created in the first frame considering the spatial localization of the intensity values and then, it is searched for in the following frames [27].

MeanShift tracker (MS) is the most popular kernel tracking algorithm due to its simplicity and efficiency. The standard version is implemented by representing the regions with color histograms. Then, the histograms are associated and finally the best candidate is located in each frame. The use of color histograms is motivated by its robustness to scaling, rotation and partial occlusion [44]. Nevertheless, MeanShift is not a perfect algorithm and its performance decreases when some of the target features appear also in the background (clutter or similar objects scenarios).

For avoiding the above-mentioned limitation, several extensions have been proposed. The Corrected Background Weighted Histogram (CBWH) presented in [45] allows to deal with clutter in the background. Not only that, but it also provides great tracking results even if the target is not properly initialized. Moreover,, the Scale and Orientation Adaptive MeanShift Tracking algorithm (SOAMST) added the ability to track objects that change its scale and orientation (for example, a vehicle moving further away from the camera and therefore, changing its size). It tries to emulate the ability Particle Filter algorithm to deal with target changes (e.g, scale, appearance) while using a deterministic approach.

Currently, the Particle Filter has been proposed for performing probabilistic video tracking where the video tracking estimation is non-linear and non-Gaussian. Opposed to use of a fixed search area for locating the target in the deterministic approaches, it proposes to sample the search space (e.g., target location) trying to guess the movement and size of the target. Among existing approaches, Color-based Particle Filter (CPF) is widely used due its easy implementation and understanding.

3.2 Deterministic video tracking

3.2.1 Template Matching

Template matching is one of the simplest techniques used in digital image processing. Its applications are numerous, including (but not limited to) edge detection, face recognition systems, medical image processing... as well as tracking [27]. The main idea of this tracking algorithm is to define the image that will be consider as a template. Then, this image is located in each frame by comparing pixel intensities. The algorithm can be described by the following steps:

1) Localization and selection of the target. A template is created with the information from the first frame. This can be done using specific object detector or manually provided (e.g, using ground-truth information)

2) Then, within a loop that covers each frame, the algorithm calculates the SSD (sum of squared differences of the image). The SSD sign is reversed and normalized to range [0,1] and can be viewed as a squared Euclidean distance. The SSD in the position (i, j) is:

$$d(i,j) = \sum_{x=i}^{i+Tx} \sum_{y=j}^{j+Ty} (I_1(x,y) - I_2(x-i, y-j))^2 \tag{3.1}$$

where I_1 is the image for searching the template (e.g., the whole frame), I_2 is the template, Tx and Ty are the width and height of the template image, and i and j range from 0 to the width and height of the image respectively.

3) Afterward, the point (i, j) with maximum SSD is found, with will be considered the center of the target.

4) After the box around this point (i, j) is created using the same size of the template, the algorithm starts the loop again in order to locate the target in the next frame.

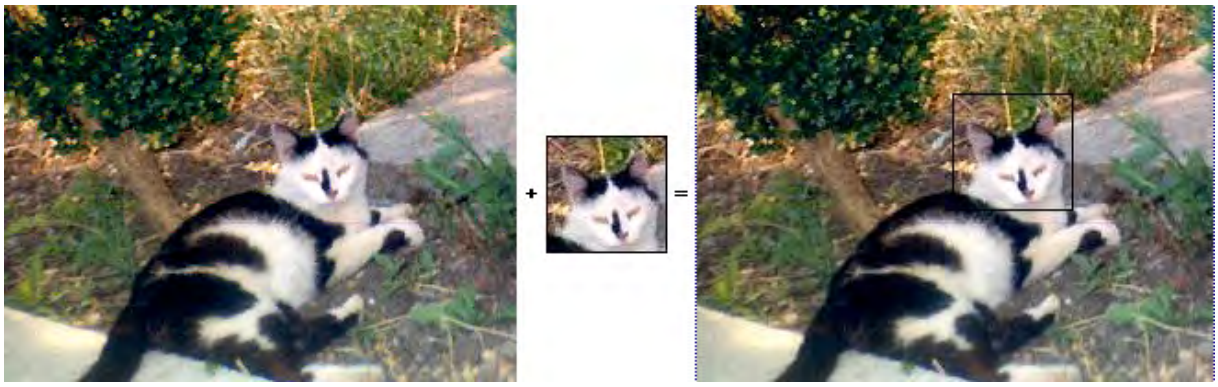


Figure 3.1: Example of TM algorithm

Brief analysis Even though this algorithm is fast and simple, there are several limitations that have to be taken into account when using it for tracking. Since the algorithm searches for an area equal to the template created by means of the color of each pixel, it does not consider changes in illumination or size, occlusion or rotation of the target. Moreover, the search is restricted to locating the maximum SSD value and, therefore, there is no possibility of estimating the size of the target.

3.2.2 MeanShift

MeanShift was originally proposed by Fukunaga and Hostetler in 1975 [46] as a generic clustering algorithm. For its use in tracking, the main idea is to construct histograms of both the target and the candidate regions including information of a certain feature (e.g, color). Later on, both histograms are compared and their similarity is measured (using, for example, the Bhattacharyya coefficient). Here, we briefly review the algorithm presented in [44].

3.2.2.1 Target representation

In the next formulas, $\hat{q}$ is the normalized histogram of the target model, and $\hat{p}\,(y)$is the target candidate:

$$\hat{q} = \{\hat{q}_u\}_{u=1...m} \tag{3.2}$$

$$\hat{p}(y) = \{\hat{p}_u(y)\}_{u=1...m} \tag{3.3}$$

Target model and target candidate $\hat{q}_u$ is the probability of the u^{th} element in the target model (normalized histogram). When x_i^* represents the normalized pixel locations in the region defined as the target model and $k(x)$ is a Epanechnikov kernel as depicted in Figure 3.2 that assigns smaller weights to pixels farther away from the center of the target, the probability of the feature $u = 1...m$ in the target model is computed as:

$$\hat{q}_u = C \sum_{i=1}^{n} k\left(\|x_i^*\|^2\right) \delta\left[b(x_i^*) - u\right] \tag{3.4}$$

where δ is the Kronecker delta function, b is a function that associates the pixel at location x_i^* to the index $b(x_i^*)$ of its bin in the quantized feature space and C is a constant defined by:

$$C = \frac{1}{\sum_{i=1}^{n} k\left(\|x_i^\star\|^2\right)} \tag{3.5}$$

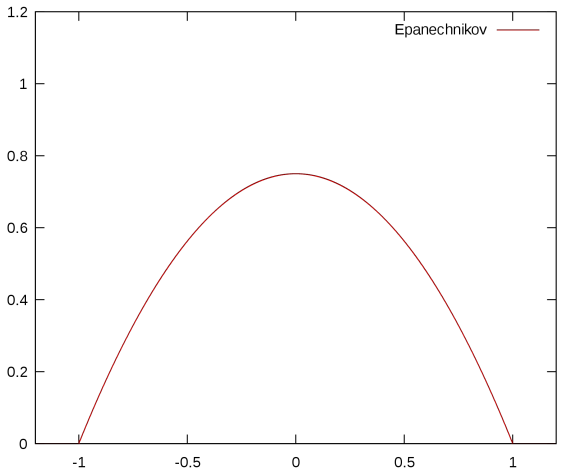


Figure 3.2: Epanechnikov Kernel: $k(x)$

The target candidate $\hat{p}(y)$ is obtained similarly to the target model.

3.2.2.2 Bhattacharyya coefficient

As a measure of similarity between the model and the candidate, the distance or coefficient of Bhattacharyya is calculated:

$$d(y) = \sqrt{1 - \rho\left[\hat{p}(y), \hat{q}\right]} \tag{3.6}$$

where the sample estimate of the Bhattacharyya coefficient between p and q is:

$$\hat{\rho}(y) \equiv \hat{\rho}\left[\hat{p}(y), \hat{q}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u} \tag{3.7}$$

3.2.2.3 Target localization

Distance minimization The goal is to minimize the distance in Equation 3.6 is equivalent to maximize the coefficient of Equation 3.7, which, if the target candidate does not changes drastically from the initial (a condition easily met in consecutive frames), can be approximated linearly and reorganiced as proposed in [44]:

$$\rho\left[\hat{p}(y), \hat{q}\right] \approx \frac{1}{2}\sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{y}_0), \hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k\left(\left\|\frac{y - x_i}{h}\right\|^2\right) \tag{3.8}$$

where

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\widehat{p}_u\left(y\right)}}\delta\left[b\left(x_i\right) - u\right] \tag{3.9}$$

and

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)} \tag{3.10}$$

where$i$ represents the number of frame and h represents the bandwidth, (i.e., the number of pixels considered in the localization process).

In order to minimize the distance, and since the first term of Equation 3.8 is not dependent of y, the second term has to be maximized. The kernel is recursively moved from the current location $\hat{y}_0$ to the new location $\hat{y}_1$ according to the relation

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)} \tag{3.11}$$

where $g(x) = -k'(x)$, being $k'(x)$ the derivative of the Epanechnikov kernel and assuming it exists for all $x \in [0, \infty)$ except for a finite set of points.

3.2.2.4 Algorithm

The tracking algorithm proposed in [44] can be summarized in the following steps:

1) The tracking algorithm is initialized, meaning that a model is created based on the first

frame. That includes defining the target location (with the information contained in the ground-truth data) as well as creating the target model by calculating the normalized histogram.

2) Then, the algorithm enters in a loop to process each frame. Given the target model $\hat{q}_u$ and the position in the previous frame $\hat{y}_0$. For each iteration, the candidate model is obtained (by creating a weighted histogram of the candidate region), and the weights are calculated as shown in Equation 3.9.

3) These weights are backprojected to each of the pixels in the candidate window. The result image is known as backprojection image and contains the information that determines the probability that each pixel has of belonging to the target as seen in Figure 3.3.

4) The new centroid is calculated and the new location is found according to Equation 3.11.

5) The convergence of the algorithm is studied: while the Bhattacharyya coefficient between the histogram of the new location and the one of the model is smaller than the one between the older location and the model, the new location is re-calculated by using the semi sum of the new location ($\hat{y}_1$) and the previous location ($\hat{y}_0$):

$$\begin{aligned} \text{While} \quad & \rho\left[\hat{p}(\hat{y}_1), \hat{q}\right] < \rho\left[\hat{p}(\hat{y}_0), \hat{q}\right] \\ \text{Do} \quad & \hat{y}_1 = \tfrac{1}{2}(\hat{y}_o + \hat{y}_1) \\ & \text{Evaluate} \quad \rho\left[\hat{p}(\hat{y}_1), \hat{q}\right] \end{aligned}$$

6) If after these operations MeanShift is converging ($\|\hat{y}_1 - \hat{y}_0\| < \epsilon$, being ϵ a set threshold), the algorithm returns to step 2 and continues with the next frame. Otherwise, the candidate model is reinitialized ($\hat{y}_0 = \hat{y}_1$) and the algorithm returns to step 3 until the new center is found.

Brief analysis MeanShift is one of the fastest tracking algorithms. It is robust when dealing with small occlusions, camera movement or blurring. On the other hand, it is not capable of dealing well with scale changes or significant changes in the object's features (such as big illumination changes or occlusions). Since it uses histograms, it does not contain information regarding spatial structure. Moreover, the use of color information is limited in presence of clutter (e.g., another object with similar colors appears next to the target the algorithm might mix them up).

3.2.2.5 Adaptive MeanShift

The adaptive MeanShift algorithm is a natural extension from the previous algorithm. In order to achieve a robust tracker for gradual changes (such as illumination changes), the target model

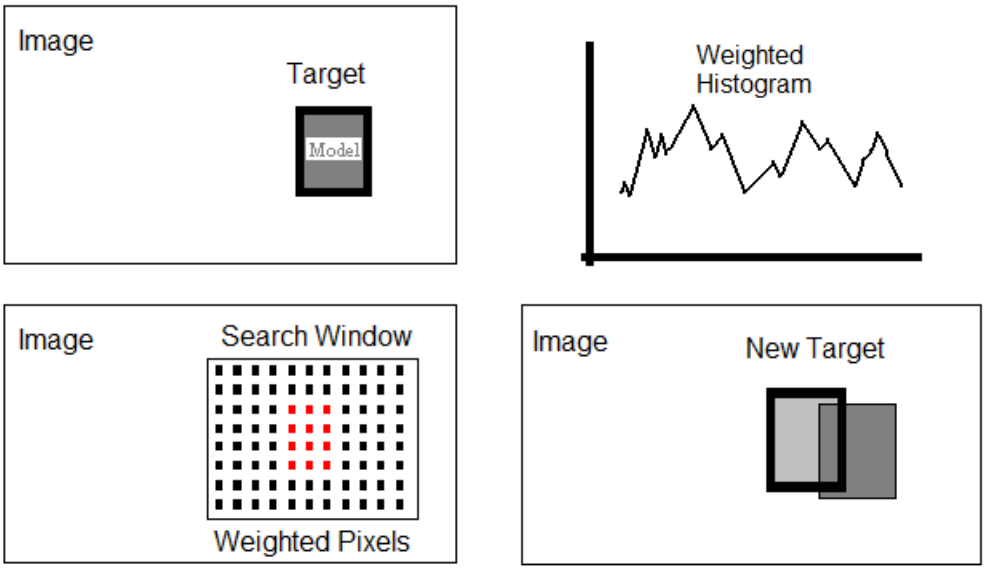


Figure 3.3: Steps of MS algorithm

$\hat{q}$ is computed for each k frame:

$$\hat{q}_k = \begin{cases} \hat{q}_{u_0} & MeanShift \\ \hat{q}_{u_k} & Adaptative\ MeanShift \end{cases} \tag{3.12}$$

The rest of the algorithm is identical to MeanShift.

3.2.3 Corrected Background Weighted Histogram

One of the disadvantages of the MeanShift algorithm is the fact that when the background contains features similar to those in the target, the tracking algorithm may lose the desired object (target). Comaniciu et al [44] proposed a new weight assignment method so that this problem could be solved by an elimination of the background information in the initialization of the target.

Unfortunately, as studied in [45], this calculations yielded a BWH (Background Weighted Histogram) proportional to the one without the new additions, and since MeanShift is invariant to a scale transformation of the weights, this method was proven to not work. The tracking result with the new weight was exactly the same as it was before the modification including the background weighted histogram.

3.2.3.1 Target model

A Corrected Background Weighted Histogram was proposed in the same article [45], allowing to correctly assign different, non-proportional weights to the pixels belonging to the background with the target model and therefore, providing a good target representation in situations with high cluster in the scene.

In this case, the modified template model was created according to Equation 3.13:

$$\hat{q}'_u = \frac{\hat{q}_u \times \hat{v}_u}{C_n} \tag{3.13}$$

where $\hat{q}_u \times \hat{v}_u$ is the multiplication of the bins of the target model histogram$\hat{q}_u$ and $\hat{v}_u$ is defined in [44] as:

$$\left\{ v_u = min\left(\frac{\hat{o}^\star}{\hat{o}_u}, 1\right)\right\}_{u=1...m} \tag{3.14}$$

and C_n is a normalization constant $C_n = \hat{q}_u \cdot (\hat{v}_u)^T$, obtained as the product of a row vector and column vector.

In Equation 3.14, $\{\hat{o}_u\}_{u=1...m}$ is the discrete representation (histogram) of the background in the feature space and $\hat{o}^\star$ is the smallest nonzero entry. The background is obtained by expanding the area of the template and considering a box twice the size of the original target model. The objective is to determine which binds of the target model histogram are similar to those of the background.

In Figure 3.4a diagram of the histograms $\hat{q}_u$, $\hat{o}_u$and $\hat{v}_u$ is depicted:

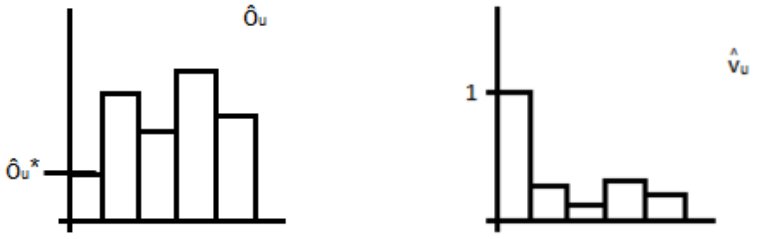


Figure 3.4: Diagram of $\hat{o}_u$and $\hat{v}_u$

The transformation in 3.13 diminishes the importance of those features which have low $\hat{v}_u$, i.e., are prominent in the background. The weights in this algorithm can therefore be written as:

$$w''_i = \sqrt{\frac{\hat{q}'_{u'}}{\hat{p}_{u'}(y)}} \tag{3.15}$$

In Figure 3.5 and Figure 3.6 a representation of how the histograms for a synthetic sequence and a simple sequence look like. In the first line there is the representation for the correct

annotation (no extra background): in (a) we find the target model, in (b) we find the target model histogram with MS, and in (c), the corrected target model histogram with CBWH.

In the second line we find depicted the scenario where the annotation was incorrect and included 50% of background. Again, (d) shows the target model, in (e) we find the target model histogram with MS and finally, (f) depicts the target model histogram with CBWH.

In 3.5 there is a clear difference between MS and CBWH when the object is not properly initialized, as can be seen by the comparison of images (e) and (f). Whereas the MS target model histogram (e) shows high values around bin 0 (black color), the CBWH target model histogram has discarded those values as part of the background and does not take those bins into account, providing a histogram very similar to the one where the initialization was properly done (c).

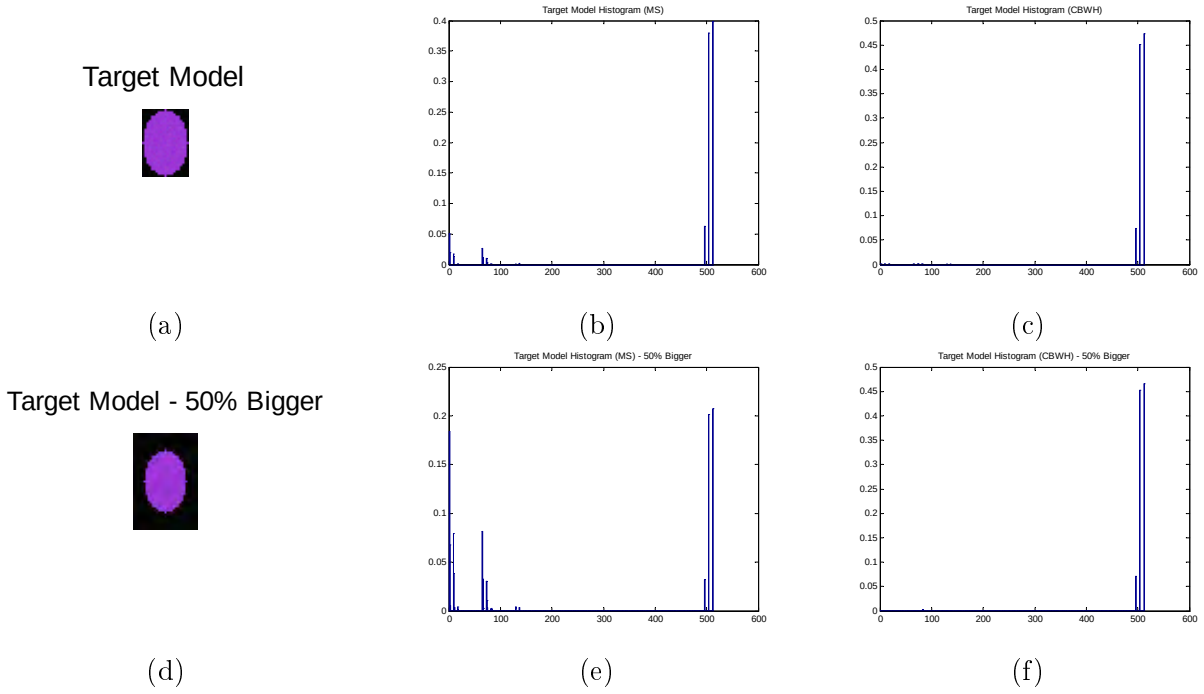


Figure 3.5: Histogram comparison with different initializations for a synthetic sequence. Data represented corresponds to (a) (d) appearance of the target, (b) (e) its histogram with MS and (c) (f) its histogram with CBWH

In Figure 3.6 the target model histogram with CBWH when the initialization is properly done (c) shows better (i.e., higher) results for the bins. However, when the initialization is not correct, the results are the same with both MS and CBWH (e) and (f), meaning that in this case, CBWH does not behave better. A reason for the decrease in this performance might be the fact that the target has different colors and is not completely uniform, therefore making it more difficult to separate background colors from target colors.

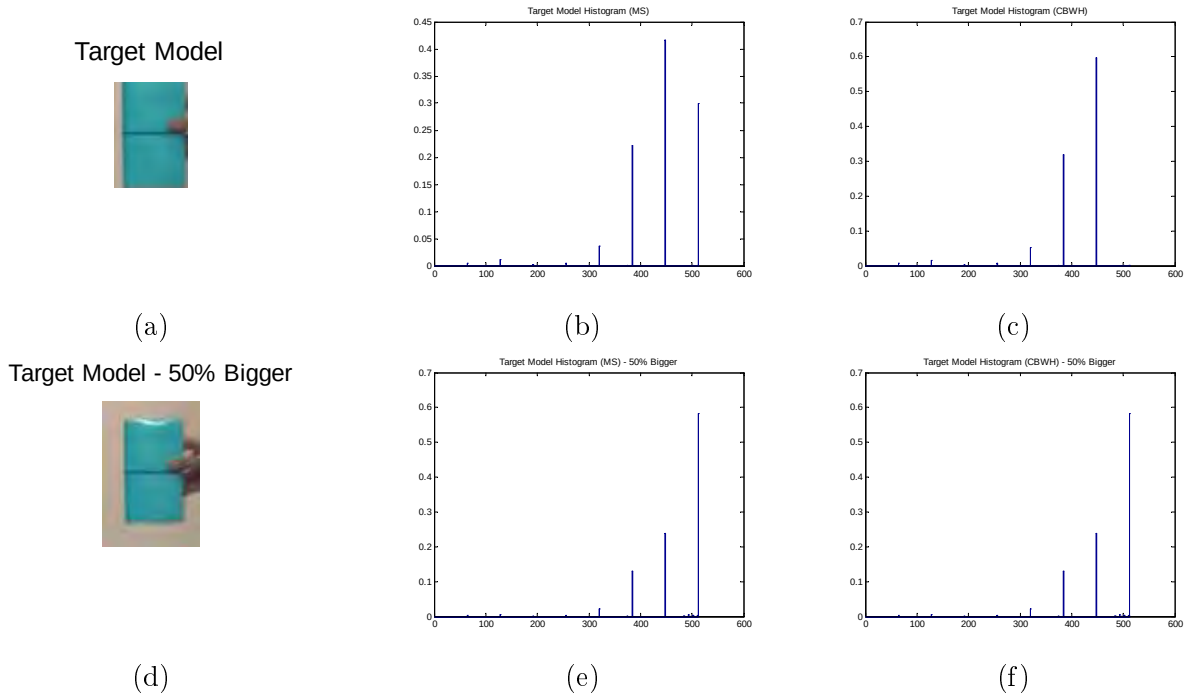


Figure 3.6: Histogram comparison with different initializations for a simple sequence. Data represented corresponds to (a) (d) appearance of the target, (b) (e) its histogram with MS and (c) (f) its histogram with CBWH

3.2.3.2 Algorithm

The algorithm described in [45] contains the following steps:

1) The first step is to create the target model according to Equation 3.4, the background-weighted histogram $(\{\hat{o}_u\}_{u=1...m})$ and $\{v_u\}_{u=1...m}$ according to Equation 3.14 and then obtain the transformed target model $\hat{q}_u$ as shown in Equation3.13. The position y_o of the target region is initialized from the previous frame.

2) Set $k = 0$.

3) Calculate the target candidate model $\hat{p}\,(y_o)$ in the current frame.

4) Calculate the CBWH weights according to Equation 3.15.

5) Calculate the new position of the candidate region using Equation3.11.

6) Update values of $d = \|y_1 - y_0\|$, $y_o = y_1$ and $k = k + 1$. Being ξ_1 the error threshold, ξ_2 the background model update threshold and N the maximum number of iterations, enter a loop.

(a) If $d < \xi_1$ or $k > N$
Calculate $\{\hat{o}'_u\}_{u=1...m}$ and $\{v'_u\}_{u=1...m}$ based on the tracking result from the current frame.
If ρbetween $\hat{o}_u$ and $\hat{o}'_u$ is smaller than ξ_2 , then $\{\hat{o}_u\}_{u=1...m} = \{\hat{o}'_u\}_{u=1...m}$ and $\{v_u\}_{u=1...m} = \{v'_u\}_{u=1...m}$ and $\hat{q}'_u$ is updated by Equation 3.13.
Stop iteration, go to step 2.

(b) Otherwise: go to step 3.

Brief analysis As previously mentioned, this algorithm was created for dealing with the situation of similar objects to the target in the background, therefore, the main advantage is that it is more robust to clutter than MS. Also, it provides good results even if the target is not properly initialized in the first frame and thanks to the faster convergence, the tracking algorithm is faster than MS. The rest of the limitations of MS also apply to this algorithm.

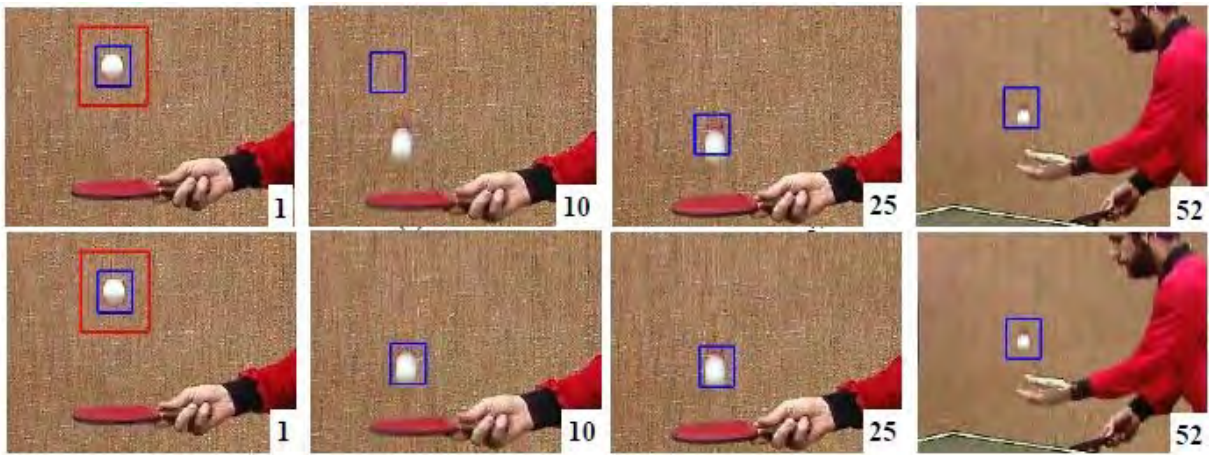


Figure 3.7: Example of MS (top row) and CBWH (bottom row). Frames shown correspond to the table tennis sequence of MPEG7 content set[45] .

3.2.4 SOAMST

The Scale and Orientation Adaptive MeanShift Tracking algorithm proposed in [47] is intended to address the problem of scale and orientation changes. As previously stated, MeanShift tracking estimates the position of the target but not its scale nor orientation. The difference between CAMSHIFT [48] and SOAMST is that while the former uses the weight image determined by the target model, the later employs the weight image derived from the target model and the target candidate model in the target candidate region in order to estimate the target scale and orientation.

3.2.4.1 Estimating the target area

The weighted area of the target in the candidate region can be viewed as the the zeroth order moment.

$$M_{oo} = \sum_{i=1}^{n} w(x_i) \tag{3.16}$$

where x_i are the pixels in the target candidate region centered at y , and $w(x_i)$ are the weights assigned to those pixels.

Using Equation 3.16 and since the Bhattacharyya coefficient is an indicator of similarities (the lower the coefficient, less features from the target are in the candidate region), the estimated accuracy of taking M_{oo} as the target area is directly proportional to the Bhattacharyya coefficient. Hence, the target is estimated as follows:

$$A = c(\rho)M_{oo} \tag{3.17}$$

where $c(\rho)$ is a monotonically increasing function (obtained empirically in [47]) with regards to the Bhattacharyya coefficient $\rho(0 \leq \rho \leq 1)$. The proposed expression of $c(\rho)$ in [47] is:

$$c(\rho) = exp(\frac{\rho - 1}{\sigma}). \tag{3.18}$$

When ρ decreases (meaning that the candidate model is not identical to the template), M_{oo} will be much bigger than the target area since $c(\rho)$ is smaller than 1, A will not be biased by containing too much information from the background. When the target gets lost (ρ tends to 0), $c(\rho)$ will be very small, and therefore A is close to 0.

In Table 3.1 the tracking results regarding three different scenarios is presented: in the first case, the estimated region contains more information from the background area than the target, creating a bigger estimation error that decreases in comparison with M_{oo}. In the second case, the information from the target area is bigger than the information from background, and therefore a smaller error is introduced (which also decreases and becomes smaller than the one with M_{oo}). Lastly, the third case is a scenario where there is no background area information (meaning that the target is perfectly initialized and the Bhattacharyya coefficient is therefore 1), and therefore no error is introduced and the video tracking is correctly performed.

Real Target area		100		150		240	
Background area		140		90		0	
Bhattacharyya coefficient		0.6454		0.7906		1	
Estimated area A under	M_{oo}	150	+50%	195	+30%	240	0%
different σ and the	$\sigma = 1.5$	118.42	+18.42%	169.59	+13.06%	240	0%
relative estimation	$\sigma = 1$	105.22	+5.22%	158.16	+5.44%	240	0%
error (%) in	$\sigma = 0.8$	96.29	-3.71%	150.09	+0.06%	240	0%
comparison with M_{oo}	$\sigma = 0.5$	73.81	-26.19%	128.28	-14.48%	240	0%

Table 3.1: Area estimation of the target with CBWH (extracted from [47])

3.2.4.2 Moment features in MeanShift tracking

The moments of the weight image can be calculated as:

$$M_{pq} = \sum_{i}^{n_h} w_i x^p y^q \tag{3.19}$$

where pair (x, y) is the coordinate of pixel i in the candidate region.

And now, the second order center moment can be described as:

$$\mu_{20} = M_{20}/M_{00} - \overline{x}^2 \tag{3.20}$$

$$\mu_{11} = M_{11}/M_{00} - \overline{xy} \tag{3.21}$$

$$\mu_{02} = M_{02}/M_{00} - \overline{y}^2 \tag{3.22}$$

where $(\overline{x, y})$ represents the centroid of the target candidate region, and this expressions can be rewritten as a covariance matrix used to estimate the width, height and orientation of the target:

$$Cov = \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \tag{3.23}$$

3.2.4.3 Estimating the width, height and orientation of the candidate

The Equation 3.23 can be decomposed as:

$$Cov = U \times S \times U^T = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix} \times \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix}^T \tag{3.24}$$

where λ_1^2 and λ_2^2 are the eigenvalues of Cov. The orientation of the two main axes of the real target in the target candidate region can be expressed as $(u_{11}, u_{21})^T$and $(u_{12}, u_{22})^T$. This is done by finding the angle (θ) between the axes.

If the axis of the ellipse used to represent the target are a and b (semi-major and semi-minor axis respectively), $a = k\lambda_1$ and $b = k\lambda_2$, where

$$k = \sqrt{\frac{A}{\pi\lambda_1\lambda_2}} \tag{3.25}$$

and therefore:

$$a = \sqrt{\frac{\lambda_1 A}{\pi\lambda_2}} \tag{3.26}$$

$$b = \sqrt{\frac{\lambda_2 A}{\pi\lambda_1}} \tag{3.27}$$

Finally, the covariance matrix becomes:

$$Cov = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix} \times \begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix}^T \tag{3.28}$$

3.2.4.4 Determining the search window in next frame

Now that the location, scale and orientation of the target are estimated, we need to determine the location of the target in the next frame. In order to do so, the covariance matrix is modified so that a increase of $\triangle d$ pixels of the target region is made. This increase means that the next search region will be bigger and therefore will have a higher probability of including the target.

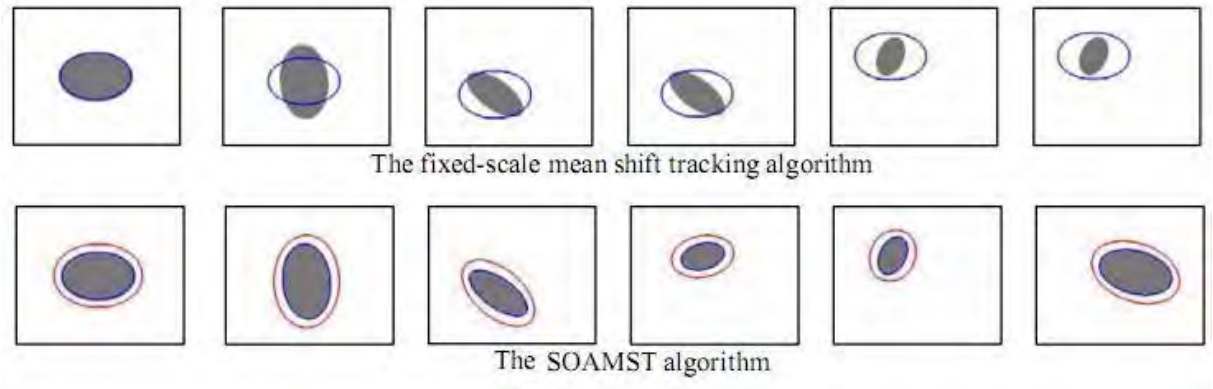


Figure 3.8: Tracking with SOAMST algorithm of a synthetic video sequence. Blue ellipses represent estimated target region, red ellipses represent target candidate region. Frames 1, 20, 30, 40, 50, 60, 70.[47]

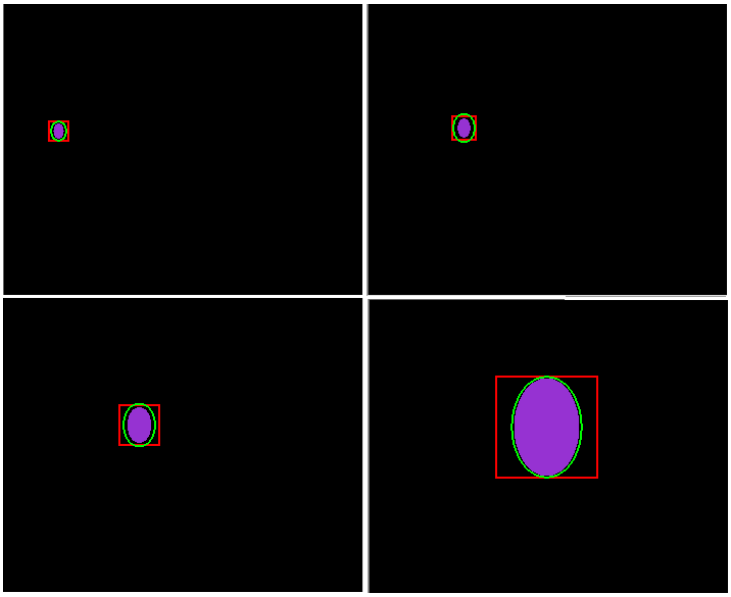


Figure 3.9: Tracking with SOAMST algorithm of a synthetic video sequence with scale changes. Green ellipses represent estimated target region, red boxes represent target candidate region. Frames 1, 20, 40 and 60.

3.2.4.5 Algorithm

The first 4 steps from the MS algorithm are performed also in the SOAMST algorithm. After that, the calculations to estimate the width, height and orientation of the target candidate model are performed using Equation 3.28. Therefore, the algorithm can be detailed as:

1) The first step is to initialize the algorithm by calculating the target model $\hat{q}$ and initialization of the position of the target y_o.

2) Set the number of iterations $k = 0$.

3) Calculate the candidate model in the current frame $\hat{p}(y_0)$.

4) Calculate and assign the weights $\{w_i\}_{i=1...n}$ with Equation 3.9.

5) Calculate the new position of the target y_1 with Equation 3.11.

6) Update values of $d = \|y_1 - y_0\|$ and $y_o = y_1$. Being ξ the error threshold and N the maximum number of iterations, enter a loop.

(a) If $d < \xi_1$ or $k \geq N$
Stop and go to step 7.

(b) Otherwise
$k = k + 1$
Go to step 3.

7) Estimate width, height and orientation from the candidate model using Equation3.28.

8) Estimate the initial candidate model for next frame adding $\triangle d$ pixels to the candidate window.

Brief analysis As previously mentioned, this algorithm was created in order to deal with the problem that arises when MeanShift is trying to follow an object which changes its scale or rotation. The main disadvantage (other than the ones that are implied from MeanShift) is the fact that it is slower than the previous algorithms.

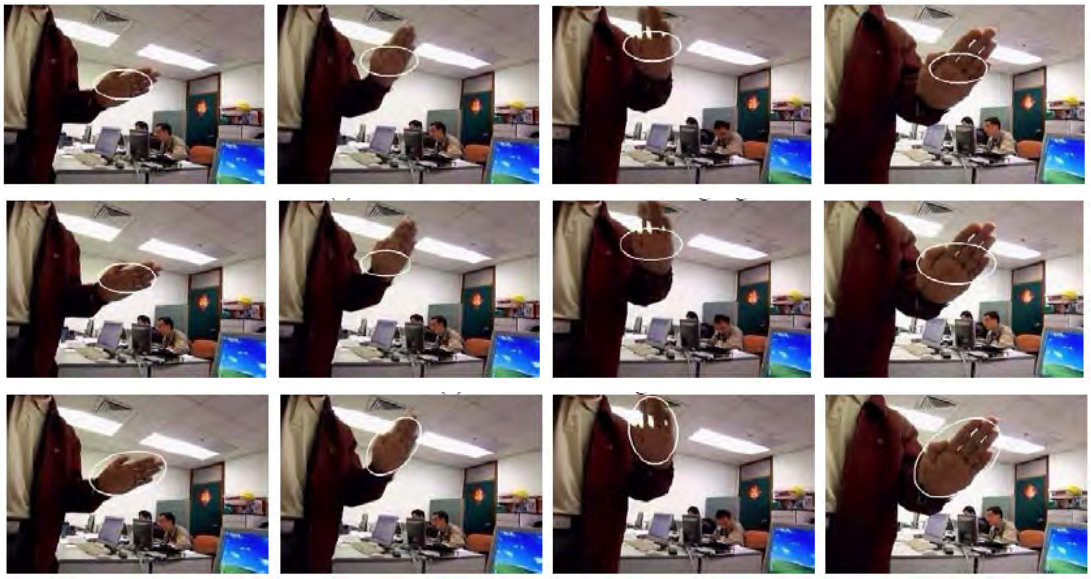


Figure 3.10: Comparison of MeanShift (upper row), Camshift (medium row) and SOAMST (lower row)[47]

3.3 Probabilistic video tracking

3.3.1 Particle Filter framework

The Particle Filter [23] is employed in order to estimate the state of a system that changes with time. This algorithm, also known as SIS (Sequential Importance Sampling) or Condensation Algorithm comprises a set of particles (samples) representing the possible states in the space and a set of weights associated with each particle, representing a posteriori values of the f.d.p.

The fundamental idea in this algorithm is to represent the posterior density by a set of random particles with associated weights and then compute estimates based on this data.

The goal of the particle filter is to estimate the state x_k given observations y_k [49], and the optimal estimate is given by:

$$\hat{x}_k = E\left[x_k | Y_o^k\right] \tag{3.29}$$

where Y_o^k is the sequence of observations up to frame k.

For video tracking, the steps for the Particle Filter are:

1) Model: a model is created using a color histogram (as in the MS algorithm)

2) Initialization of the samples[1]: to start the target tracking, the Particle Filter creates a set of random points over the image. At this point, the particles can be created randomly or employing some a priori information (target's size, approximate position, etc)

3) Prediction: once the samples of the previous frame are created, a small modification in the state is computed, for example, adding noise that will contribute to the variability of the system. This will help to estimate the state of the target in the current frame.

4) Update: each sample gets an assigned weight depending on the similarity with the target model. The data of the current frame is used to compute this similarity.

5) Sampling: this steps allows to get rid of particles with low weights, speeding up the process and discarding non-useful data. The set of samples is designed for the analysis of the next frame.

Once the sampling stage is finished, this process is repeated from 3 to 5 until the end of the sequence is reached.

3.3.2 Color-based PF algorithm

The Particle Filter tracking algorithm is implemented based on the description of [50]. For each time step, the output of the filter is the particle (or sample) set $X_t = \left\{(x_t^{(n)}, \pi_t^{(n)})\right\}_{n=1,\ldots,N}$ of N weighted particles, where each particle $x_t^{(n)}$ represents one hypothetical state of the target weighted by $\pi_t^{(n)}$. Each particle $x_t^{(n)}$ at time t is defined by the following parameters:

$$x_t^{(n)} = (x, y, H_x, H_y, \theta) \tag{3.30}$$

where x and y represent the position of the target ellipse, H_x and H_y are the semi axes of the elliptic region and θ is the angle of the target ellipse (we have omitted the t sub indexes of these parameters for clarification purposes).

[1] *Samples* and *particles* are synonyms in this explanation

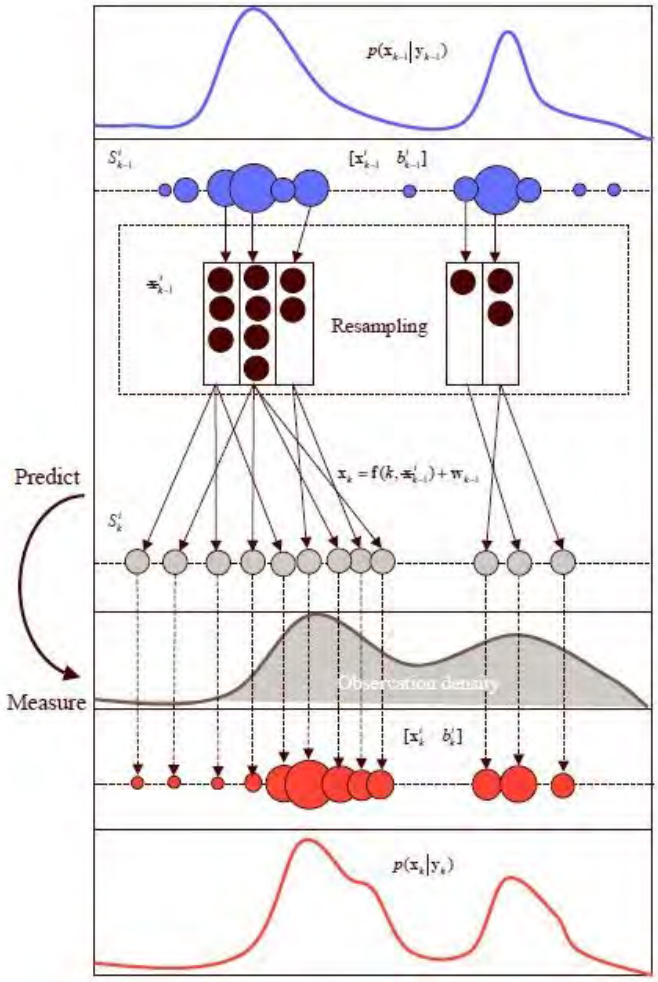


Figure 3.11: The particle filter from [49]

Given the particle set X_{t-1} and the target model q:

$$x_t = Ax_{t-1} + z_{t-1} \tag{3.31}$$

where x_t is a particle (sample) of the distribution representing an ellipse, A defines the deterministic component of the first order model to describe the movement at a constant velocity of a region and z_{t-1} is a multivariate Gaussian random variable (probabilistic component).

An iteration for the color-based Particle Filter is detailed below:

1) Propagate each particle from the new set by a linear stochastic differential equation:

$$s_t^{\prime(n)} = As_{t-1}^{\prime(n)} + z_{t-1}^{(n)} \tag{3.32}$$

2) Calculate color distribution with:

$$p_{x_t^{(n)}}^{(u)} = f \sum_{i=1}^{I} k \left(\frac{\left\| x_t^{(n)} - x_i \right\|}{a} \right) \delta \left[h(x_i) - u \right] \tag{3.33}$$

for each particle of the set X_t, then calculate Bhattacharyya coefficient (again, for each particle of the set X_t) and weight each particle of the set X_t with:

$$\pi_t^{(n)} = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{\left(1-\rho\left[p_{x_t^{(n)},q}\right]\right)}{2\sigma^2}} \tag{3.34}$$

where σ is the variance of the Gaussian. In this case, the kernel is:

$$k = \begin{cases} 1 - r^2 & : r < 1 \\ 0 & : otherwise \end{cases} \tag{3.35}$$

although an Epanechnikov Kernel could also be used.

3) Estimate the mean state of the set X_t:

$$E\left[X_t\right] = \sum_{n=1}^{N} \pi_t^{(n)} x_t^{(n)} \tag{3.36}$$

4) Resampling: select N particles from the particle set X_{t-1} . This selection is done based on the weight of each particles, according to a function that eliminates samples with lower weight such as:

$$x_{t-1}^{\prime(n)} = f(x_{t-1}^{(j)}, \pi_t^{(n)}) \tag{3.37}$$

Lastly, and update of the set is done by finding the smallest probabilities within a threshold.

Brief analysis Since this implementation employs multiple state hypotheses and a model of the system dynamics, it provides a robust framework to model uncertainty. And since less likely objects are eventually removed from the tracking process, particle filters can deal with short occlusions. Some of its disadvantages are the high computational complexity (as the number of particles increases) as well as the difficulty of optimally estimating the number of particles.

Figure 3.12: Example of the color-based particle filter algorithm for a sequence with occlusions (frame 19) and scale changes (frames 31 and 81) [50]

Chapter 4

Proposed Evaluation Protocol

This chapter presents the proposed protocol for evaluating video tracking algorithms. First, the motivation for designing an evaluation protocol for tracking is given in section 4.1. Later, an overview of the protocol and the covered issues are described in section 4.2, followed by a definition of some of the most common issues in video tracking in section 4.3. The complete dataset including those issues (with different levels of complexity) is detailed in section 4.4. Finally, the aspects analyzed in the protocol are presented in section 4.5.

4.1 Introduction

As studied in section 2.4, one of the issues that appear when dealing with the creation of a tracking algorithm is the lack of appropriate datasets with both sequences and annotations for its evaluation. Moreover, several metrics exists being not clear which one should be used to evaluate tracking.

In this context, we propose an extensible methodology to evaluate video tracking algorithms. It provides a complete evaluation framework including the test video sequences (dataset), the set of evaluation metrics and the aspects to understand the advantages and drawbacks of the tracking algorithm under evaluation.

4.2 Evaluation framework

The proposed framework for evaluating video trackers is depicted in Figure 4.1. As it can be observed, it is composed of two stages: tracking analysis and performance evaluation.

4.2.1 Tracking Analysis

Represents the tracking algorithm that we want to evaluate.

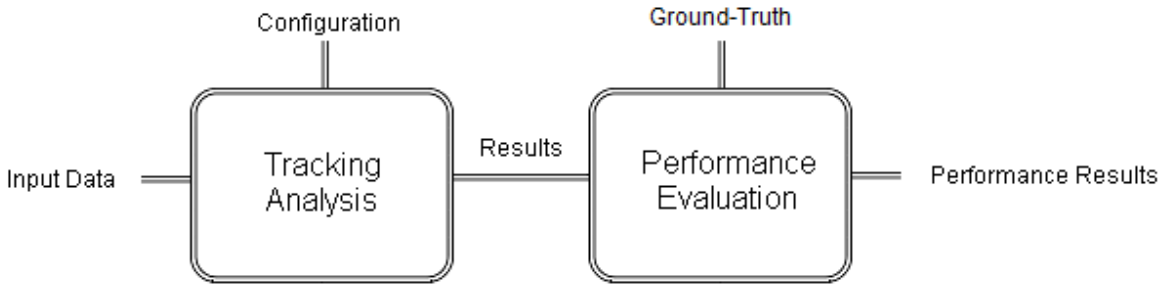


Figure 4.1: Proposed evaluation framework for video object tracking.

- Input Data: video data to be analyzed. In order to evaluate the accuracy of the algorithm, appropriate data have to be used considering the tracking task (e.g., video sequences with noise, clutter, illumination changes, complex movement, etc).
- Configuration: includes all the parameters of the tracking algorithm that the user can manually set (e.g., window search).
- Results: a file is generated with the tracking data containing information of the position and size of the target. It has a specific format (the same as the ground-truth file) in order to apply the protocol, and is one of the two input files of the Performance Evaluation stage.

4.2.2 Performance Evaluation

Describes the evaluation of the generated tracking data.

- Tracking results: the results file as described above.
- Ground-Truth: this file contains the annotations regarding the true location of the target throughout the whole sequence. There is one row for each frame of the video, and eight columns corresponding to the frame number, the target id, the target label, the position (x and y) of the target center, the half-axis (Hx and Hy) of the ellipse that fits the area of the target and the angle of this ellipse.
- Performance Results: these results are obtained by comparing the ground-truth and the results files and are later stored in a plain text file and a csv file.

4.3 Modeled problems

For designing a dataset to evaluate video tracking, several issues have to be taken into account corresponding to a number of problems that can arise in real-world conditions. Although some

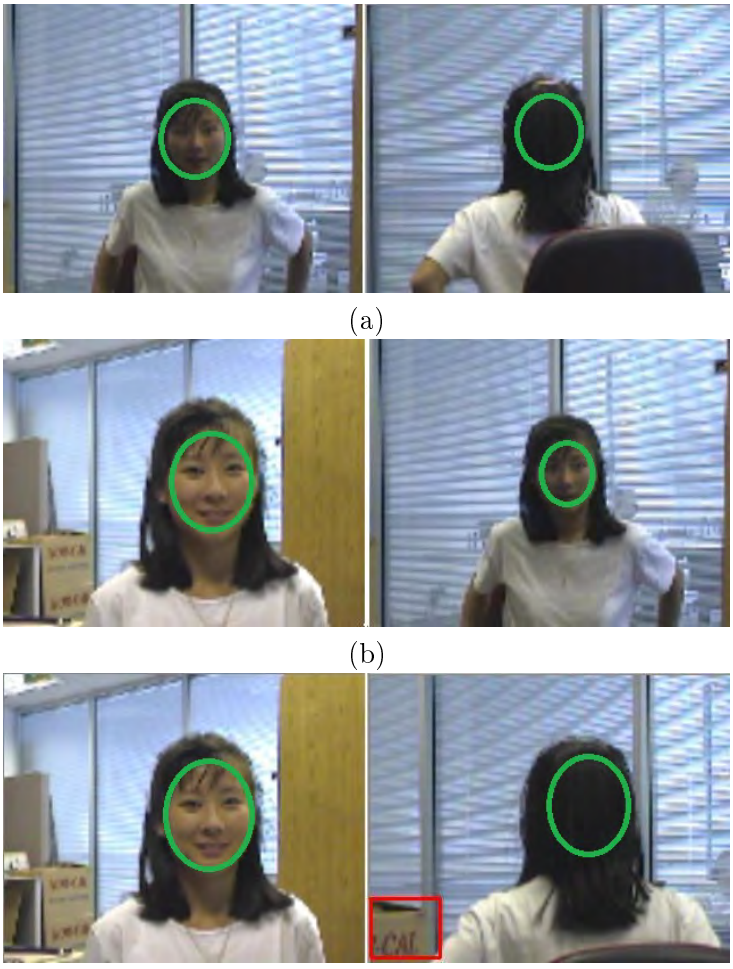


Figure 4.2: Example of tracking issues: (a) Appearance change, (b) Scale changes and (c) Similar objects in background. Target annotations are represented by means of a green ellipse and similar objects are indicated with a red box.

issues may not be easily noticeable for the human eye (such as slight changes in the intensity of the pixels), cameras are able to capture them. Hence, they have to be considered when the tracker is evaluated. The following issues have been modeled in the dataset: complex movement, gradual and abrupt illumination changes, noise, occlusion, scale changes, and similar objects. An example of some of the mentioned issues is depicted in Figure 4.2. .

4.3.1 Complex or fast movement

When the target changes its trajectory unexpectedly or increases its speed abruptly, the tracker might lose it for a few frames (and sometimes even more). In most of existing trackers, this issue is overcame by increasing the search area (or the prediction strategy); the bigger it is, the less

this issue will affect the tracker performance. However, execution time is also increased. The factor to measure these sequences is the speed increase of the target.

4.3.2 Gradual illumination changes

If a sequence is long enough the illumination might change due to weather conditions, time passing, etc. In this case, the whole illumination of the scene changes gradually and globally (i.e., affecting the whole image). The factor to measure gradual illumination changes is the pixel intensity difference between low and high illumination conditions for each sequence.

4.3.3 Abrupt illumination changes

When the target changes its position and enters in a differently illuminated area the tracker might be confused and lose the target. This issue is very common in real sequences (i.e., when a target enters in a shadowed area that presents different illumination or, when tracking indoors, lights go on/off suddenly). The factor to measure abrupt illumination changes is the difference between the average illumination of the two areas that present different illumination.

4.3.4 Noise

Noise appears as random variations over the values of the pixels of the image and can significantly degrade the quality of the extracted features. Hence, tracking performance might be affected. In this case, we model different noise level by adding white Gaussian noise to the image with different mean values.

4.3.5 Occlusion

An occlusion is defined when an object gets between the camera and the target. It can be partial (only part of the target is occluded) or total (where the whole target is occluded for at least 1 frame). The factor to measure occlusions is the percentage of the target occluded by other objects.

4.3.6 Scale changes

As targets can move at variable distances to the camera, scale changes have to be considered (specially for tracking in videos that correspond to large areas). A scale change happens when a target moves across a scene and increases or decreases its size due to changes in its distance from the camera. The factor to measure scale changes is the percentage of size change experienced by the target in the last frame with regards to its size in the initialization frame.

4.3.7 Similar objects (clutter)

If there are objects similar in the feature spaces considered in the target model (e.g., color, texture, edges), the tracker might be confused by these objects and start following a wrong target that has similar characteristics to those of the target. The factor to measure scenes with similar objects in the background was the degree of complexity in each sequence, which was obtained subjectively[1].

4.3.8 Configuration issues

Another set of issues is the one that concerns the configuration of the tracker. Not only the sequences are the source of tracking errors: if the system is not properly designed and parameters are not properly selected, the tracker's performance will decrease. Some of this issues are, for example, the selection of the optimum algorithm's parameters (such as search window size, number of particles, etc) or the correct initialization of the target.

4.4 Dataset

The selection of the test scenarios is one of the most important steps when developing an evaluation protocol. Each previously mentioned issue has to be represented in the dataset for achieving a correct understanding of the capabilities of the tracking algorithm. Moreover, different levels of complexity have to be covered in the test data. Hence, this dataset is designed with four complexity levels including both real and synthetic sequences. They are described as follows.

4.4.1 Level 1: Synthetic Sequences

For the first level, we use synthetic sequences that have been artificially generated to address a particular tracking problem. These sequences are interesting because they provide testing conditions in a controlled environment allowing to isolate the issues. However, algorithms are not expected to perform in a similar way for real-world sequences.

In order to do so, simple scripts have been composed using MATLAB[2] to generate such sequences. These scripts allow to define parameters in each sequence such as speed of target, size of occlusive objects, illumination change speed (gradual vs. abrupt), noise level, etc. A summary of the generated sequences is listed in Table 4.1. Sample frames are shown in Figure 4.3.

[1] The degree of complexity was obtained by analyzing the sequences and taking into account several factors such as the number of similar objects in the scene, the closeness to the target and the amount of frames where similar objects appeared.

[2] These scripts are based on MATLAB code provided by the VPULab.

Problem	Sequence	Criteria	Factor	Frames
Complex Movement	S1	The position of the object changes randomly with a **speed** factor (changing in each sequence)	10%	100
	S2		20%	100
	S3		30%	100
	S4		40%	100
	S5		50%	100
Gradual Illumination	S1	The illumination of each sequence changes gradually until a **maximum intensity difference** is reached in each sequence	0%	100
	S2		25%	100
	S3		50%	100
	S4		75%	100
	S5		100%	100
Abrupt Illumination	S1	Half of the image in each sequence has a different pixel intensity (added to the original image) until a **maximum intensity difference** is reached in each sequence	10%	100
	S2		20%	100
	S3		30%	100
	S4		40%	100
	S5		50%	100
Noise	S1	A white Gaussian noise with different **mean** in each sequence is added to the original sequence	0.01	100
	S2		0.02	100
	S3		0.03	100
	S4		0.04	100
	S5		0.05	100
Occlusion	S1	A second object exists in the scene occluding a **percentage** of the target	15%	100
	S2		50%	100
	S3		85%	100
	S4		100%	100
	S5		100%	100
Scale Changes	S1	The target increases its size throughout each sequence until a **maximum size** (depending on the original size) is reached	150%	100
	S2		225%	100
	S3		350%	100
	S4		650%	100
	S5		1100%	100
Similar Objects	S1	An object similar to the target with a different size in each sequence increases the **degree of complexity**	10%	100
	S2		30%	100
	S3		60%	100
	S4		80%	100
	S5		90%	100

Table 4.1: Description of Level 1 sequences

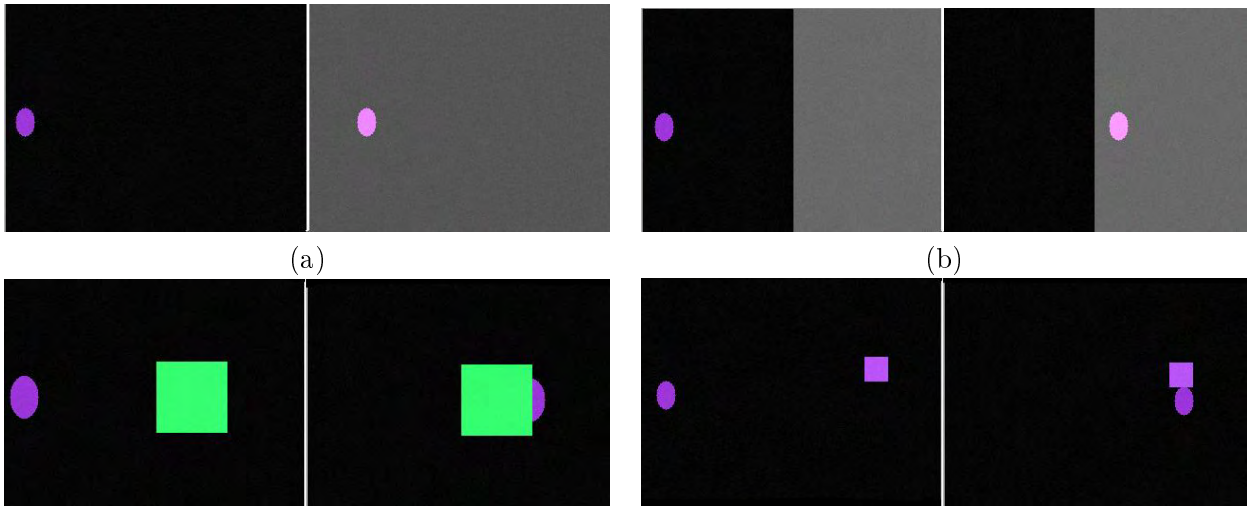


Figure 4.3: Synthetic sequences of Level 1 with the following issues: (a) Gradual Illumination Change (GradualIllumination_S5), (b) Abrupt Illumination Change (AbruptIllumination_S5), (c) Occlusion (Occlusion_S5), (d) Similar Objects in Background (SimilarObjects_S3).

4.4.2 Level 2: Simple Real Sequences

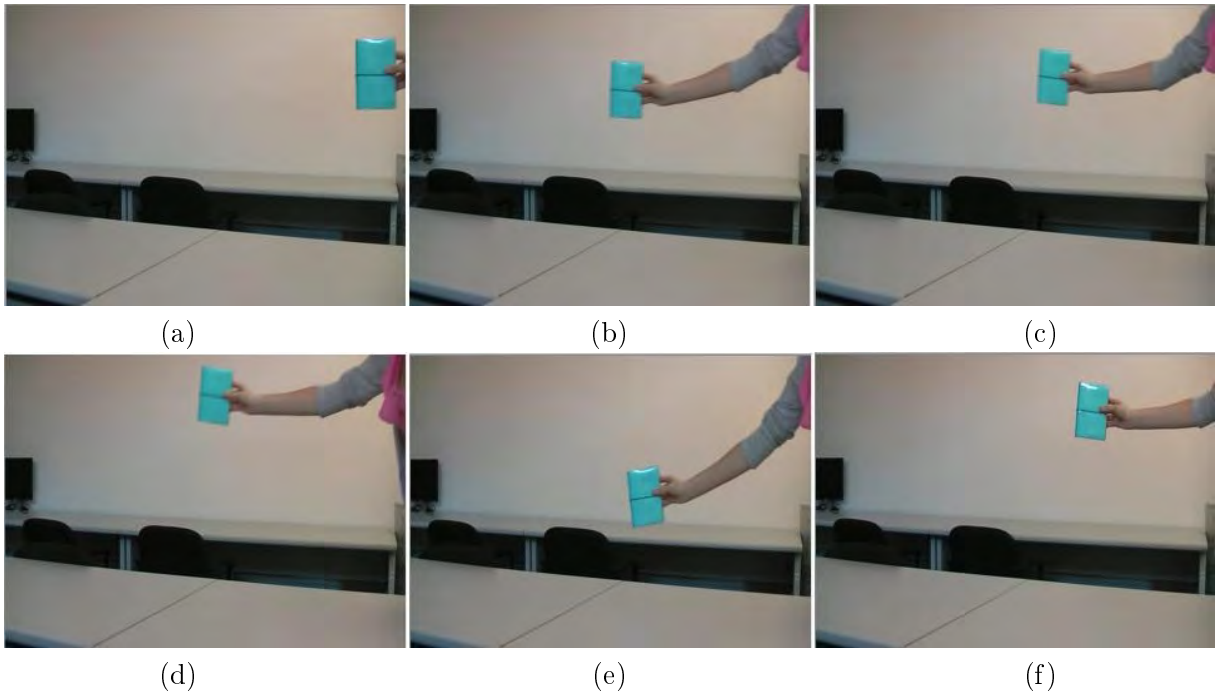
These sequences provide the second level of the evaluation protocol being the natural extension of the synthetic ones detailed in previous sub-section. These sequences, recorded with a Panasonic HDC-HS9 High Definition camera, represent real testing conditions in a laboratory environment defining controlled situations. The issues are as isolated as possible. Afterward, the annotation for each sequence was created. A summary of these sequences can be found in Table 4.2.

4.4.2.1 Basic Sequence

Sequence with no issues: the movement is simple, there is no relevant amount of noise nor illumination changes, no similar objects in background and no occlusion or scale changes. This sequence will not be used as is in the tracking evaluation, but it serves to include different noise and illumination changes to generate the test data for those issues. An example of this sequence can be seen in Figure 4.4.

Problem	Sequence	Criteria	Factor		Frames
Complex Movement	S1	The target changes its **speed** and **position** abruptly throughout the scene	50%	0%	216
	S2		75%	50%	205
	S3		100%	100%	196
Gradual Illumination	S1	The illumination of each sequence changes gradually until a **maximum amount** is reached.	25%		270
	S2		50%		270
	S3		75%		270
Abrupt Illumination	S1	Half of the image in each sequence has a different pixel intensity until a **maximum amount** is reached	20%		270
	S2		30%		270
	S3		40%		270
Noise	S1	A white Gaussian noise with different **mean** in each sequence is added to the original sequence	0.01		270
	S2		0.02		270
	S3		0.03		270
Occlusion	S1	A second object exists in the scene occluding a **percentage** of the target	<50%		321
	S2		>80%		350
	S3		100%		375
Scale Changes	S1	The target increases its size throughout each sequence until a **maximum size** is reached	240%		454
	S2		330%		430
	S3		600%		500
Similar Objects	S1	The target moves in front of a similar object for different lengths of time, with different **degrees of complexity**	30%		150
	S2		60%		280
	S3		90%		315

Table 4.2: Description of Level 2 sequences



(a) (b) (c)

(d) (e) (f)

Figure 4.4: Example of Level 2, basic sequence, frames 0, 50, 100, 150, 200 and 250 from the basic sequence

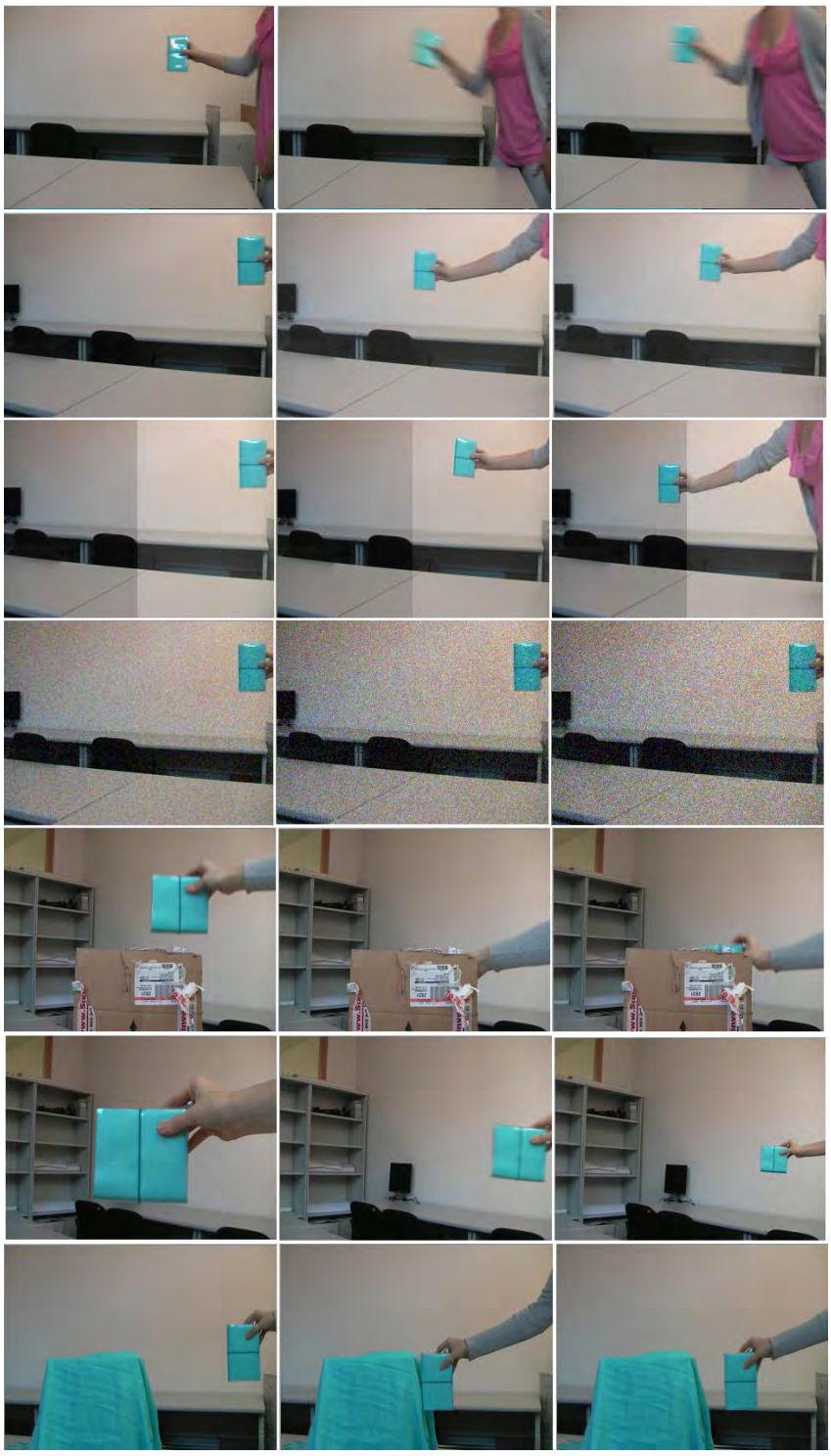


Figure 4.5: Examples of Level 2 sequences including (from top row to bottom row): complex movement, gradual illumination changes, abrupt illumination changes, noise, occlusion, scale changes and similar objects.

Figure 4.6: Example of a Level 3, Cars sequence, where the target is the black car in the front at the left of the image that encounters similar objects as it moves to the left side of the image.

4.4.3 Level 3: Complex Real Sequences

The third level of the evaluation dataset includes sequences from previously existing datasets. Most of them have been captured in non-controlled scenarios without containing any external modification (e.g., artificial noise), containing issues as isolated as possible. Since most of the videos are too complex and include several issues in the same scene, sequences were cut in clips to avoid the effect of various issues in the same sequence. As no information was provided for the issues appearing in each sequence, this step requires a carefully study of the existing datasets. Afterward, the ground-truth file for each final clip was annotated.

These sequences are grouped in three categories according to the type of target as seen in Table 4.3: cars (obtained from MIT Traffic dataset [51] and Karlsruhe Cars dataset [52]), faces (obtained from TRECVID 2009 [53], CLEMSON dataset [54] and VISOR [55]) and people (obtained from PETS 2009 [56], TRECVID 2009 [53], i-Lids [57], CAVIAR [58] and PETS 2000 [59]). As each target presents different characteristics, the tracking algorithm should be studied depending on the type of the target. In the first set of videos the target is a car that is not easy to annotate (i.e., the target model contains a high amount of background data) and it is composed by few different color regions as seen in some examples in Figure 4.6. In the second set, the target is the face (or head) of a person, where the target is easy to annotate (i.e., the target model contains a low amount of background data) and only a few different color regions represent the target as seen in Figure 4.7. Finally, the third set includes sequences where the target is a whole person that is difficult to annotate (i.e., the target model contains a high amount of background data) and it is composed by several color regions as seen in 4.8.

A detailed description of the selected test sequences is available at Appendix C.

4.4.4 Level 4: Multiple Issues Sequences

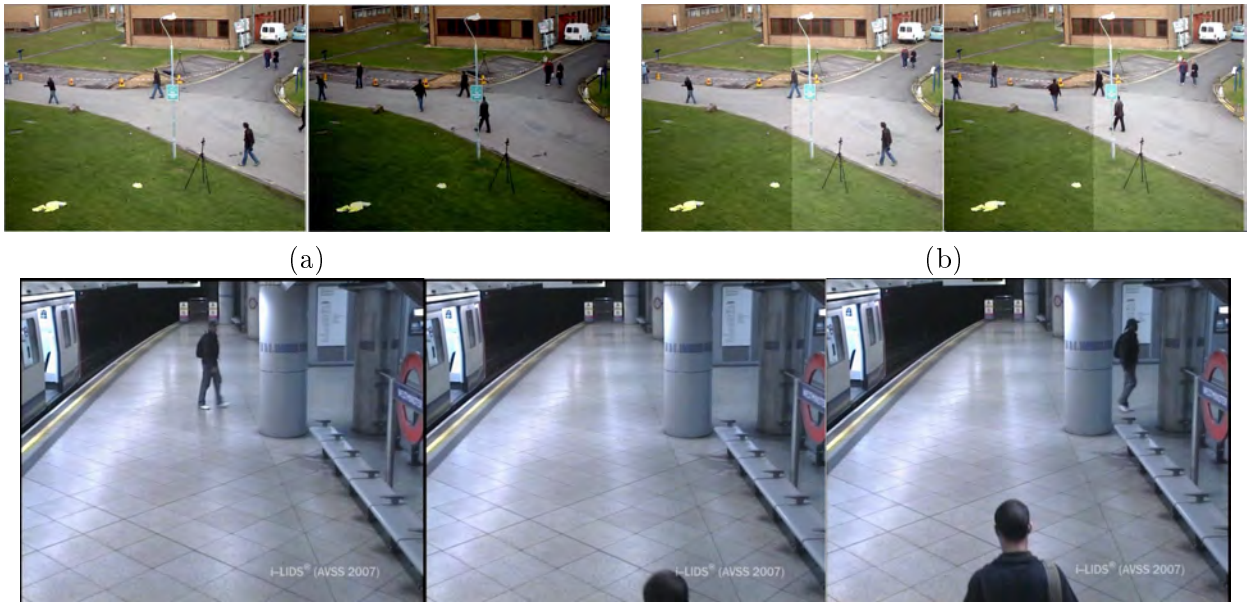
This level contains the most complicated sequences, which are clips from other datasets that include several issues. The subdivisions are the same as the ones detailed for Level 3 sequences. Once the algorithms are tested for each issue individually, it is a good idea to check the per-

Problem	Sublevel	Sequence	Criteria	Factor	Frames
Complex Movement	Faces	S1	The target changes its **speed** abruptly throughout the scene	90%	83
	People	S2		30%	77
		S3		60%	69
		S4		90%	58
Gradual Illumination	Faces	S1	The illumination of each sequence changes gradually until a **maximum intensity difference** is reached in each sequence	30%	100
		S2		60%	100
		S3		90%	100
	People	S4		30%	60
		S5		60%	60
		S6		90%	60
Abrupt Illumination	Cars	S1	Part of the image in each sequence has a different pixel intensity until a **maximum intensity difference** is reached in each sequence	30%	200
		S2		60%	300
		S3		90%	350
	Faces	S4		30%	100
		S5		60%	100
		S6		90%	100
	People	S7		30%	60
		S8		60%	60
		S9		90%	60
Noise	Cars	S1	Sequences include natural noise (snow) or a white Gaussian noise manually added to the original sequence with different **means**	30%	200
		S2		0.01	75
		S3		0.02	75
		S4		0.03	75
	Faces	S5		0.01	100
		S6		0.02	100
		S7		0.03	100
	People	S8		0.01	60
		S9		0.02	60
		S10		0.03	60
Occlusion	Cars	S1	Objects in the scene occlude a **percentage** of the target	30%	132
		S2		60%	77
		S3		90%	116
	Faces	S4		30%	170
		S5		60%	99
		S6		90%	304
	People	S7		30%	54
		S8		60%	158
		S9		100%	156
Scale Changes	Cars	S1	The target increases or decreases its size throughout each sequence until a **maximum size** regarding the original size is reached.	60%	165
		S2		60%	281
		S3		30%	282
	People	S4		90%	130
		S5		90%	98
		S6		60%	435
		S7		90%	134
Similar Objects	Cars	S1	An object similar to the target appears on the scene near the target (**degree of complexity**)	30%	313
		S2		90%	65
		S3		60%	195
	People	S4		90%	240
		S5		60%	154
		S6		30%	54

Table 4.3: Description of Level 3 sequences

(a) (b)

Figure 4.7: Example of Level 3, Faces sequences: (a) depicts a scenario including complex movement and (b) shows a noise sequence.



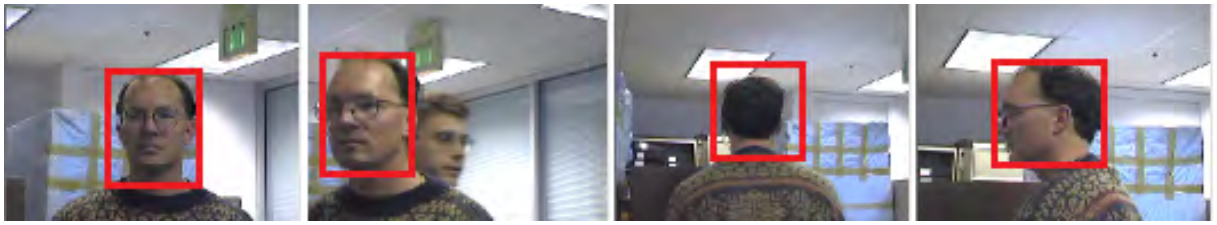
(a) (b)

(c)

(d)

Figure 4.8: Example of Level 3, People sequences: (a) and (b) depict the gradual and abrupt illumination changes scenarios (respectively), (c) shows an example of a man occluded by a column and (d) shows an example of a scale change (where the target is the man in red with a backpack).

(a)



(b)

(c)

Figure 4.9: Example of Level 4, (a) Cars sequence, (b) Faces sequence, (c) People sequence.

formance in more realistic (and complex) scenarios, such as the ones below where several issues appear in each video (as can be seen in Table 4.4). All these sequences were downloaded from the MIT Traffic dataset (cars) [51], CLEMSON dataset (faces) [54] and PETS 2009 dataset (people) [56].

In Figure 4.9 (a), the target is the red car found in the middle of the intersection in the first image. As it moves throughout the scene, there is a scale change (when it moves further away from the camera) as well as an illumination change (when in enters the shadowed area). In (b), the target is the face from the first image. In this sequence, the issues are similar objects in the background (second image, where another person is behind the target) as well as different appearance changes (first, when the target is facing opposite the camera, and then when the target looks sideways). In (c) (from the group of People in Level 4), the target is the man in black in the right side of the image. As it moves, it becomes occluded (second image), changes its size (third image) and encounters objects with similar color (similar objects in background, last image).

	Sequence	Appearance Changes	Complex Movement	Illumination Changes	Occlusion	Scale Changes	Similar Objects
Cars	020_red			⋆⋆⋆		⋆⋆	
	020_silver			⋆⋆		⋆⋆	
	020_whitevan	⋆⋆		⋆⋆		⋆⋆⋆	⋆⋆⋆
Faces	seq_bb	⋆⋆					⋆⋆
	seq_jd	⋆⋆					⋆⋆⋆
	seq_mb	⋆⋆⋆				⋆⋆	⋆⋆
	seq_ms				⋆⋆		⋆⋆⋆
	seq_sb	⋆⋆⋆	⋆⋆⋆		⋆⋆⋆	⋆⋆⋆	⋆⋆
	seq_villains2	⋆				⋆⋆	⋆⋆⋆
People	S2_L1_view001_1				⋆⋆	⋆⋆⋆	⋆⋆
	S2_L1_view001_2				⋆⋆⋆	⋆⋆	⋆⋆⋆
	S2_L2_view001_1			⋆	⋆⋆	⋆⋆	
	S2_L2_view001_2			⋆		⋆⋆	
	S2_L3_view001_1			⋆⋆	⋆⋆	⋆⋆	
	S2_L3_view001_2			⋆⋆		⋆⋆⋆	

Table 4.4: Level 4 Sequences and Issues

4.5 Performance criteria

In this protocol we consider the following performance criteria (i.e., aspects are to be evaluated): accuracy (which measures how successful the algorithm is and therefore, a certain metric capable of representing the success has to be selected from available ones as described in sub-section 5.1), stability (which measures the performance change in different runs), efficiency (which measures the execution time of the tracking algorithm) and parameters (which measures the variation of the algorithm results when the parameters are modified).

4.5.1 Accuracy Evaluation

In order to evaluate the selected algorithms, one metric was chosen: SFDA, which was detailed in depth in 2.3, but as a reminder, SFDA (Sequence Frame Detection Accuracy) calculates in each frame the spatial overlap between the estimated target location and the ground-truth annotation..

$$SFDA = \frac{\sum_{t=1}^{t=Nframes} FDA(t)}{\sum_{t=1}^{t=Nframes} \exists(N_{GT}^{(t)} OR\, N_{D}^{(t)})} \tag{4.1}$$

where

$$FDA(t) = \frac{OverlapRatio}{\frac{N_{GT}^{(t)} + N_{D}^{(t)}}{2}}, \tag{4.2}$$

and $N_{GT}^{(t)}$ and $N_{D}^{(t)}$ represent the number of ground-truth and target annotations respectively in the t^{th} frame.

4.5.2 Stability Evaluation

The same algorithm is run several times and the variance of the results obtained is calculated. In order to do so, the user manually determines the number of runs and then the whole tracking analysis is repeated. In deterministic trackers, all results should be the same when applied to the same sequence, no matter how many repetitions are done. On the other hand, when using probabilistic trackers, is expected that results vary for each execution.

4.5.3 Efficiency Evaluation

The computational cost for each sequence and tracker is obtained. Afterward, a script calculates the execution time per pixel (by dividing the total time by the number of pixels of the target) for each algorithm, providing a way of comparing which algorithm was faster when taking into account the sequences from the complete dataset.

4.5.4 Parameters Evaluation

The goal of this evaluation is to study the performance of the algorithm when selecting different values of its parameters. There are two different cases analyzed in this section: the optimum parameters and the initialization of the algorithm.

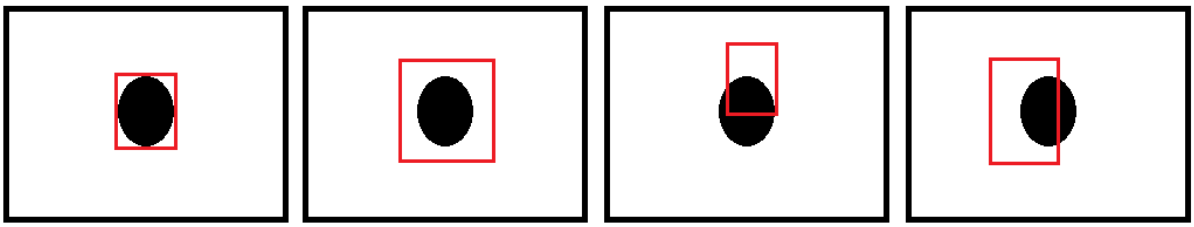
4.5.4.1 Optimum Parameters Evaluation

In the first case, several parameters (different for the deterministic and probabilistic algorithms) were chosen, as well as the test range for the values of each parameter. After running the system for each value, the accuracy results (using the SFDA measure) are calculated for every scenario. Then, the value with the best performance is selected as the optimum for the parameter.

4.5.5 Initialization Evaluation

We consider two tests for the initialization evaluation. First, an evaluation of how the initialization affects each algorithm and issue was performed. Later, an evaluation of how different initializations affect different targets was performed.

In order to do so, a MATLAB script was used to modify the ground-truth information to compose three sets of files for each sequence. The files were identical to the original ground-truth except for the initialization in the first frame. The first set of files modifies the size of the target, with the only condition that the overlap between the new target box and the original one was at



(a) (b) (c) (d)

Figure 4.10: Initialization issues where (a) represents the correct initialization, (b) presents changes in size, (c) presents change in position and (d) presents changes in size and position.

least 80%, 50% and 30% respectively. The second set of files modifies the position (that is, the location of the center of the target) from the original one, again with the same overlap conditions as the first case. The third set of files included a modification in both size and location of the center of the target's box. The same overlap condition was met in this case as well[3]. Afterward, the algorithms were run and a measure of the accuracy (using SFDA measure) was performed.

In Figure 4.10, different initialization cases are depicted: (a) shows the correct initialization of the target, (b) shows the initialization when the size of the target is incorrect (but the location of the center of the target is correct), (c) is the case when the position is not properly set (and the size does not change with regards to the original), and (d) depicts the scenario when both size and position are inaccurate.

4.5.5.1 Initialization errors vs. Issues

In the first case, the idea is to measure the performance of the algorithms when the initialization is not perfectly accurate, and how a modification in this initialization affects each issue. The total number of sequences selected for this test was 31 (approximately, 4-5 sequences per issue), all with a medium degree of complexity. Three different tests were performed for each initialization error (size, location, and a combination of both), adding up to a total of 9 different files for each sequence.

4.5.5.2 Initialization errors vs. Targets

In this case, the idea is to measure extensively how different errors in the initialization affect the targets in general. In order to do so, 6 sequences (2 from cars, 2 from people and 2 from faces) were selected from the Level 3 dataset, all with a medium degree of complexity. Afterward, 60 different files were created for each type of initialization error, adding up to a total of 180 different files for each sequence.

[3]The creation of the modified ground-truth files for a selection of sequences from the dataset was done using the code from [2]

Chapter 5

Experimental Work

This chapter presents the experiments performed to test the evaluation protocol proposed in chapter 4. First, a study of selected metrics for performance evaluation of tracking is described in section 5.1, providing conclusions to analyze and understand later results presented in this chapter. Then, section 5.2 details the application of the evaluation protocol to the selected tracking algorithms in chapter 3, including its evaluation aspects (Stability, section 5.2.1, Accuracy, section 5.2.2, Repeatability, section 5.2.3, Efficiency, section 5.2.4).

5.1 Comparison of performance evaluation metrics of tracking

5.1.1 Global analysis

A comparison of the most representative metrics is performed as a previous step for the development of the proposed protocol. To obtain the data for evaluation, the entire dataset was analyzed with the selected algorithms: Template Matching (TM) [27], Corrected Background Weighted Histogram (CBWH) [45], MeanShift (MS) [44], MeanShift Adaptive (MSA) (modified version of [44]) and Scale and Orientation Adaptive MeanShift Tracker (SOAMST) [47]. They have been implemented in Matlab by adapting the code provided by their respective authors. Then, the mean value of the selected metrics was obtained for each algorithm result with the sequences.

The selected metrics to be compared are (as explained in section 2.3): Sequence Frame Detection Accuracy (SFDA, [39]), Average Tracking Accuracy (ATA [42]), Average Tracking Error (ATE [42]), Area Under the loss-track ratio Curve (AUCinv [7]) and Track Completeness (TC [6]). All measures range from 0 to 1, being 0 the lowest tracking performance and 1 the highest tracking performance, with the exception of ATE that presents the opposite behavior.

In Figure 5.1 represent the histograms of the results of the selected metrics. SFDA presented three distinctive peaks that correspond to low, medium and high complexity sequences, while ATE values can be also grouped into three groups less separated. Neither ATA nor AUCinv

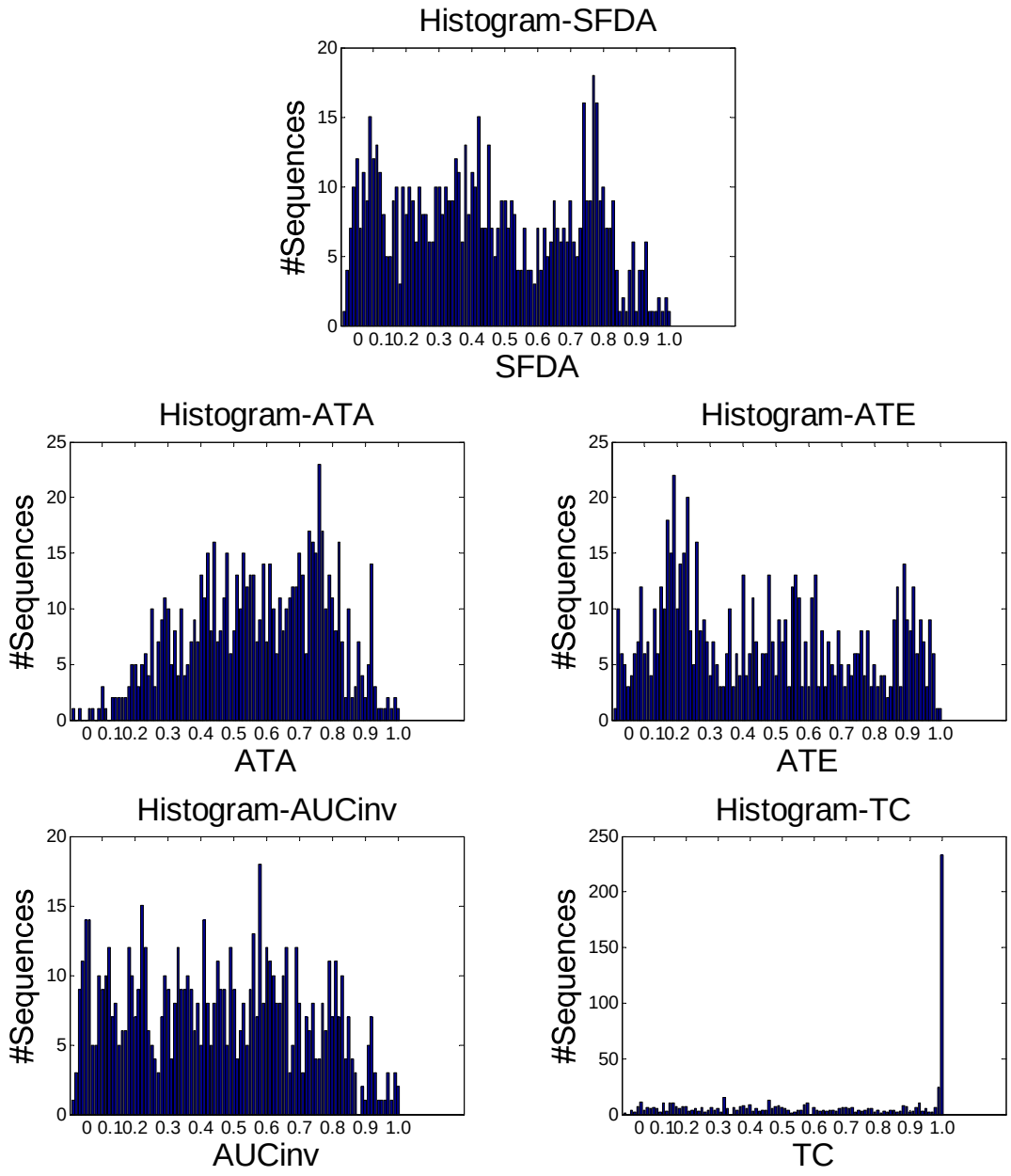


Figure 5.1: Histograms for SFDA, ATA, ATE, AUC and TC for all sequences of the dataset

provided a clear view of the behavior of the sequences nor does TC, which gave high values to most of the sequences, thus not providing an insight of the performance of the algorithms.

As seen in section 2.3.3.2, SFDA is the overlap ratio divided by the number of frames where either a tracked object or a ground-truth object is present. On the other hand, ATA is the overlap ratio divided by the number of frames where there is a mapping between the estimated target location and ground-truth annotation (overlap is higher than 0). Since it is possible that

Test Sequence	SFDA	ATA	ATE	AUCinv	PixelOv	TC
HEADTRACK_seq_bb	.318	.498	.562	.282	.271	.580
HEADTRACK_seq_jd	.264	.480	.633	.263	.258	.490
HEADTRACK_seq_mb	.480	.519	.474	.474	.477	.916
HEADTRACK_seq_ms	.383	.517	.508	.390	.380	.700
HEADTRACK_seq_sb	.310	.491	.659	.307	.309	.624
HEADTRACK_seq_villains2	.391	.403	.488	.355	.356	.845

Table 5.1: Performance evaluation of MS for test sequences with medium error.

the algorithm loses the target (and therefore, the overlap is 0), ATA (as well as ATE) proves to be an incomplete metric, while SFDA provides more information regarding the tracking results.

In the following subsections, we present a detailed analysis of the selected metrics by considering tracking results with low, medium and high accuracy (visually inspected). Specifically, we focus on their real range of values and how well they represent the tracking accuracy.

5.1.2 Detailed analysis

In this case, a selection of sequences and algorithms from the previous section was performed. We distinguish three scenarios: low, medium and high complexity. Each scenario contains a selection of sequences which were run with the following algorithms: MS (for the medium complexity scenario), CBWH (for the low complexity scenario) and SOAMST (for the high complexity scenario). The selection of algorithms was performed based on the behavior of each one.

Table 5.1 shows the results corresponding to medium tracking error. The obtained values of SFDA and ATA demonstrate the conclusions achieved for the global analysis. The relation between SFDA and ATE is obvious: the higher the success (SFDA), the lower the error (ATE). Also, the sum of both metrics combined is always lower than 1. Since SFDA can be viewed as a true positive rate and ATE as a false positive rate both combined (added) provide the complete Positive Rate which range is [0,1]. A closer look at the values of AUCInv (the inverse of the area under the curve) and PixelOv (the spatial overlap between estimated and ground-truth target) reveals that not only are both metrics very close to each other but also to the SFDA values previously studied. Therefore, these metrics (SFDA, AUCInv and PixelOv) contain the same information. In the sequence "HEADTRACK_seq_ms" results are much better than in the other sequences due to the fact that the complexity of that sequence is much lower.

For the low error scenario the algorithm was run with low complexity sequences as seen in Table 5.2. The TC allows to understand the obtained values. While an algorithm with an SFDA of 0.465 doesn't seem like a good tracker (as seen in sequence "dtneu_schnee_redcar"), the TC reveals that actually 70% of the tracking was correct. Therefore, the perfect tracking (100% of track completeness) is achieved with an SFDA slightly smaller than 0.65.

Test Sequence	SFDA	ATA	ATE	AUCinv	PixelOv	TC
dtneu_schnee_redcar	.465	.512	.380	.493	.495	0.71066
l3_cars_noise_3	.735	.735	.163	.766	.767	1
l3_cars_noise_2	.741	.741	.160	.758	.759	1
l3_cars_noise_1	.753	.753	.151	.779	.781	1
l3_faces_noise_high	.645	.645	.231	.487	.486	1
l3_faces_noise_medium	.672	.672	.211	.521	.521	1
l3_faces_noise_low	.662	.662	.218	.513	.512	1
l3_people_noise_high	.716	.716	.188	.665	.663	1
l3_people_noise_medium	.712	.712	.190	.654	.652	1
l3_people_noise_low	.703	.703	.196	.648	.647	1

Table 5.2: CBWH Metrics for the noise sequences (low error)

The high complexity set of sequences with the poor performance algorithm deliver the expected results as can be seen in Table 5.3. The relationships between metrics do not change, and the only interesting thing to notice are the TC values. While very low SFDA's would demand low TC's as well, some cases do not behave as expected. This is due to the fact that most TC values are very high (as seen in Figure 5.1). Therefore, TC can not be considered an appropriate metric. For example, as seen in Table5.2, both sequences "l3_faces_noise_medium" (SFDA=.647) and "l3_people_noise_medium" (SFDA=.744) have the same TC. This means that the Track Completeness provides high results when the object is found, but does not provide information regarding how accurately the object is found.

An example of how different SFDA values provide good performance can be found in Figure 5.2. In this figure two sequences with different SFDA's and the same TC (1) are compared. The first sequence ("HEADTRACK_seq_mb") had an SFDA of 0.480 since the tracking is good but the accuracy of the target box is not perfect, while the second sequence ("l3_people_noise_high") had a value of 0.716 (with a much better location accuracy), and both present good performance.

As a result from this analysis, the conclusion is that the SFDA metric allows a good understanding of the performance of an algorithm and therefore will be used in following sections.

Test Sequence	SFDA	ATA	ATE	AUCinv	PixelOv	TC
AB_Easy_man	.168	.260	.520	.213	.201	.509
mv2_002_redcar	.256	.359	.389	.267	.265	.612
mv2_003_blackcar	.292	.408	.432	.310	.305	.675
mv2_005_silvercar	.301	.401	.412	.308	.306	.699
visor2_man_head	.078	.524	.899	.087	.082	.125
visor5_man_head	.023	.448	.969	.037	.027	.041
visor6_man_head	.050	.259	.853	.051	.049	.106

Table 5.3: SOAMST Metrics for the occlusion sequences (high error)

Figure 5.2: Comparison of two sequences with the same TC and different SFDA

Furthermore, although the value range for SFDA is [0,1], according to these experiments, the real range of values can be established as [0,0.65] approximately.

5.2 Application of the proposed protocol

In this section we describe the application of the protocol proposed in chapter 4 to the selected algorithms of chapter 3: CBWH, MS, MSA, SOAMST, TM and PF. First the algorithms are used to track the targets annotated in the test sequences. Then, we apply the proposed evaluation protocol and to analyze their results as follows.

5.2.1 Parameters

5.2.1.1 Optimum parameter determination

First, we have determined the optimum value of the parameters of the selected tracking algorithms. Then, the optimal values are used for applying the evaluation protocol. The whole dataset was employed for this parameter optimization stage.

For deterministic tracking (CBWH, MS, MSA and SOAMST), the parameter chosen to be tuned was the increase of the search window size (with regards to the target size). Thus, several runs were made for the selected algorithms. TM uses the whole image as search area, and a reduction of its size did not provide any satisfactory effect on its accuracy. First, for CBWH, MS and MSA, the window size was increased by a percentage of 5%, 10% or 25% with regards to the original target window size. Second, for SOAMST, the search size window was incremented by 5, 10 and 15 pixels as recommended by the authors. The obtained results are listed in Table 5.4. Best results are achieved for an increase of 10% for MS and 25% for CBWH and MSA. For

	SFDA		
Increase	CBWH	MS	MSA
5%	0.51	.45	.20
10%	.53	**.51**	.24
25%	**.57**	.46	**.27**

(a)

	SFDA
Increase	SOAMST
5	**.35**
10	.33
15	.31

(b)

		SFDA															
		CBWH				MS				MSA				SOAMST			
		Size				Size				Size				Size			
		25%	50%	100%		25%	50%	100%		25%	50%	100%		25%	50%	100%	
Increase	5%	.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**	
	10%	.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**	
	25%	.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**		.33	.39	**.43**	

Table 5.4: SFDA values for varying size of search area for the deterministic algorithms. Results are shown in (a) for an increase and (b) for a decrease in the search area.

$svxy$		300				400				500				600			
$svHx, svHy$		0.5	1.0	1.5	2.0	0.5	1.0	1.5	2.0	0.5	1.0	1.5	2.0	0.5	1.0	1.5	2.0
N	200	.309	.291	.285	.288	.320	.298	.294	.294	**.330**	.299	.296	.296	.323	.308	.301	.303
	400	.321	.294	.283	.291	.325	.296	.289	.293	.321	.302	.295	.295	**.332**	.298	.296	.301
	600	.316	.287	.287	.285	.318	.291	.289	.287	.314	.302	.293	.298	**.331**	.305	.295	.298
	800	.319	.292	.286	.278	.320	.291	.291	.289	**.327**	.301	.287	.294	.320	.295	.294	.298

Table 5.5: SFDA values for varying size of search area for the probabilistic algorithm

SOAMST, best results are obtained for smaller increase (5 pixels).

For the Particle Filter algorithm, four parameters were considered for tracker stability: N (number of particles with the values 200, 400, 600 and 800), $svxy$ (variance of the x,y position with the values 300, 400, 500 and 600) and $svHx, svHy$ (variance of the major and minor axes of the ellipse with the values 0.5, 1, 1.5 and 2). Obtained results are reported in Table5.5, where best results for each row are bolded, showing that the best configuration is with higher $svxy$ (i.e., a higher variance in target position), with a small $svHx, svHy$ (i.e., tolerating small changes in target size) and with a medium N For the rest of the experiments, the values used will be: $N = 400$, $svxy = 600$ and $svHx = svHy = 0.5$.

5.2.1.2 Initialization errors vs. Issues

This experiment allows to understand the effect of inaccurate initialization on the algorithm performance for the analysis of sequences including the issues previously described. This initialization is commonly described as a bounding box defined by its center and dimensions (width and height). Sequences were selected in order to include the widest spectrum possible for each issue, therefore including sequences for every level. The results obtained are detailed in Appendix E, and a summary of these experiments is presented below.

Conclusions As depicted in Figure 5.3, algorithms are more vulnerable to changes in the size of the initialization, whereas changes in the position are easily solvable. Overall, algorithms behave well with errors in the initialization, being SOAMST the best one in this case: thanks to the dynamic approach to finding the target box, errors in the initialization do not affect the algorithm's performance and the algorithm appeared extremely robust for all errors in the

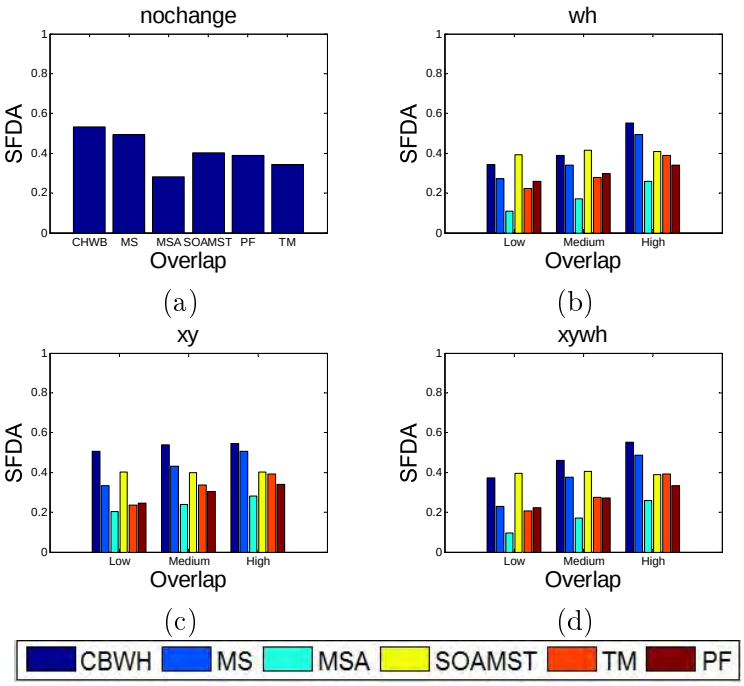


Figure 5.3: Comparison of different initialization errors for each algorithm.

initialization. More details are available in Appendix E.

5.2.1.3 Initialization errors vs. Type of targets

This test allows to understand the effect of inaccurate initialization depending on the three types of targets specified in this datasets: cars, faces and people. A selection of sequences from the dataset with medium complexity was used. The car sequences are "mv2_006_redtruck", "mv2_020_silver", the faces sequences are "HEADTRACK_seq_jd", "HEADTRACK_seq_sb" and the people sequences are "l3_people_illuminationlocal_medium", "PETS2009_S2_L2_view001_1". We perform this test similarly to the previous section. The results are detailed below, and the figures depict tracking performance with (a) no initialization errors, (b) changes in size, (c) changes in position and (d) changes in size and position of the target. The overlap indicated in each figure represents that at least 30% (Low), 50% (Medium) or 80% (High) of the modified bounding box overlaps with the ground-truth annotation.

Cars The obtained results are depicted in Figure 5.4. There is a progression that shows that the higher the overlap, the better results (as expected). In general, worst results are obtained for changes in the size of the target as expected, whereas changes in the position proved to be more easily solved. CBWH provided great behavior when changes in the position occurred as the algorithm is capable of dealing with background information better than MS or the others.

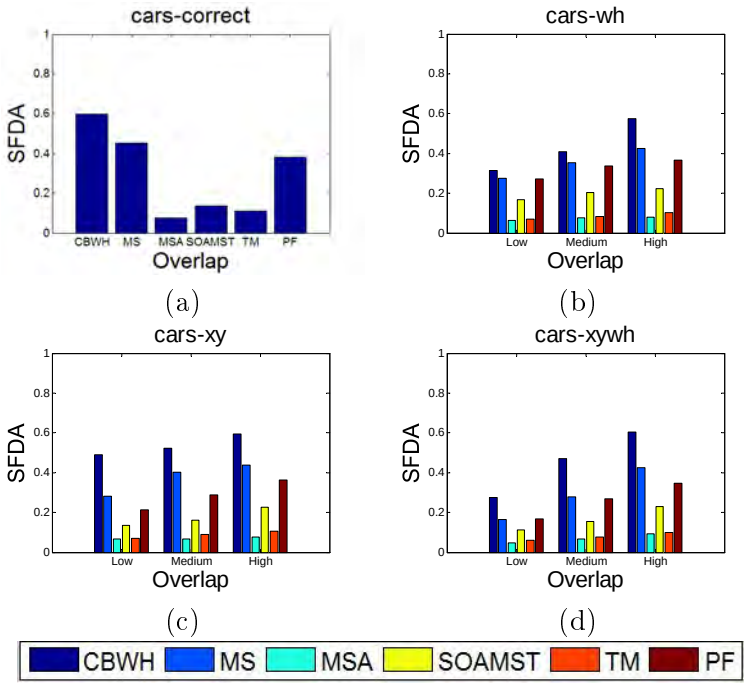


Figure 5.4: Initialization errors comparison for Cars sequences.

Faces Results are depicted in Figure 5.5. Faces sequences had a lower general complexity which is shown in these results. In general, this sequences had a better behavior with errors in the initialization, either in size, position or a combination of both. In fact, for some algorithms (e.g., MS) better results were obtained when dealing with slight errors in size. This may be due to the fact that the original ground-truth annotation was not completely accurate, and therefore was more room for improvement.

People Results depicted in Figure 5.6 show that these sequences are the most invariant to errors in the initialization, especially in the size case. SOAMST is capable of providing very good results since it can adapt to the size and position of the target, and therefore, small errors can actually improve final results. The rest of the algorithms lower their performance as the overlap decreases, as expected.

5.2.2 Accuracy

This section provides an in-depth study of the performance of each algorithm. Results are presented in terms of SFDA and are classified considering the four levels of the proposed dataset and the complexity of the test sequences for each level. Note that, as explained in section 5.1, a SFDA value of 0.65 or higher means an almost perfect tracking.

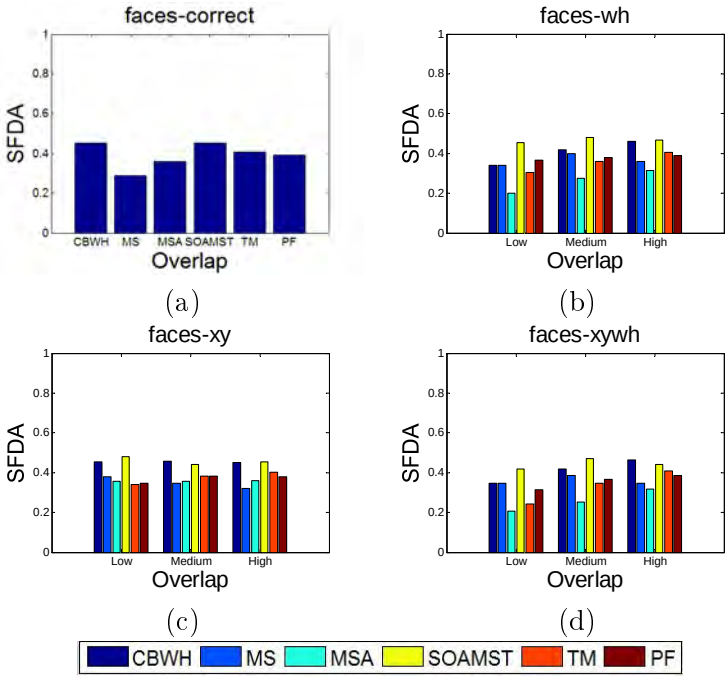


Figure 5.5: Initialization errors comparison for Faces sequences

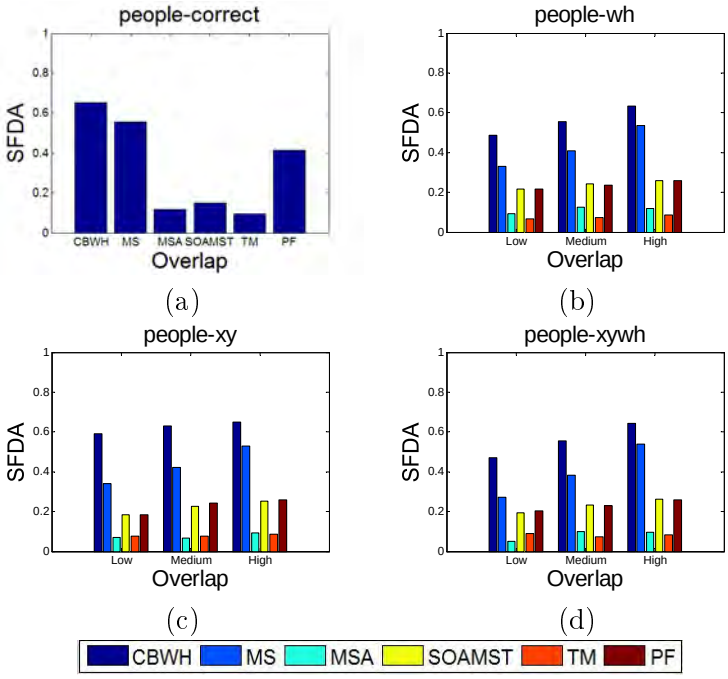


Figure 5.6: Initialization errors comparison for People sequences

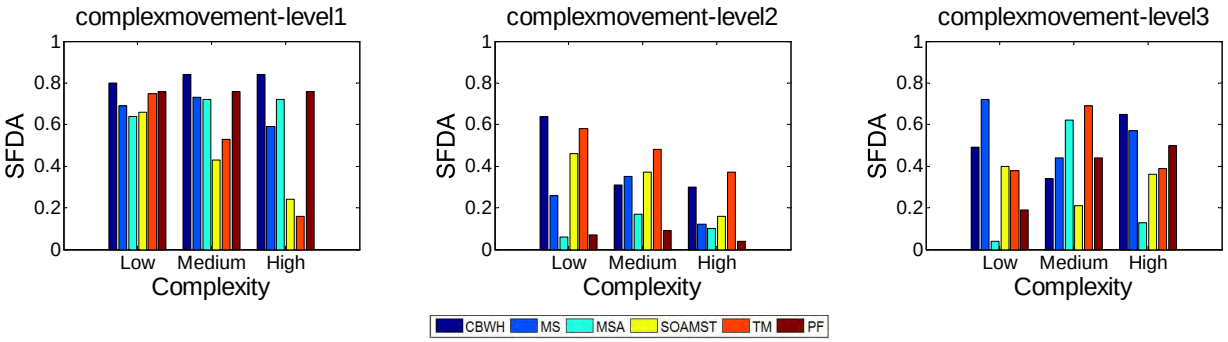


Figure 5.7: SFDA values of selected tracking algorithms for Complex Movement sequences

Complex movement Results for targets with complex movement can be seen in Figure 5.7. Globally, it can be observed that there is no relation between the different levels of the dataset.

In Level 1 (synthetic sequences) the best performance is that of CBWH. MS and MSA obtained good results as well as PF, which provided approximately the same ATA in all three degrees of complexity. SOAMST and TM had problems as the complexity level increases, with poor results for high complexity sequences.

In Level 2 (lab sequences), TM presented the best results followed by CBWH, whereas PF and MS and MSA decreased their success rates drastically. SOAMST got good results when the complexity is low, although the performance decreased as the complexity increased: this is due to the fact that the algorithm loses the target in complex videos. The overall decrease of SFDA in this level can be explained with the high complexity of the sequences.

Finally, most Level 3 (lab sequences) results do not show a clear pattern demonstrating the difficulty of estimating the complexity of the movement in real sequences. The algorithms behave erratically: while CBWH has a medium performance with a maximum for high complexity, MS had best results for low complexity sequences, and MSA had its maximum performance medium complexity sequences. Finally, SOAMST obtained good results except for medium complexity sequences (due to the fact that when the speed of the target is high, the tracking algorithm is not able to follow the target, decreasing the size of the target window and therefore affecting the SFDA values), as well as PF. An example of the execution of the algorithms for the Level 3 sequence "visor1_man_head" can be found in Figure 5.8, where all algorithms except for MSA and TM provide good results. The problem with MSA is that it changes its model in each frame, and when if the algorithms loses the target it will immediately update the model with incorrect information from the background, therefore making it very difficult to recover from errors, and due to changes in the orientation of the target, TM has problems with the template, therefore losing the target.

Figure 5.8: Example of complex movement with a Level 3 sequence. Frames: 1, 20, 40, 60. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

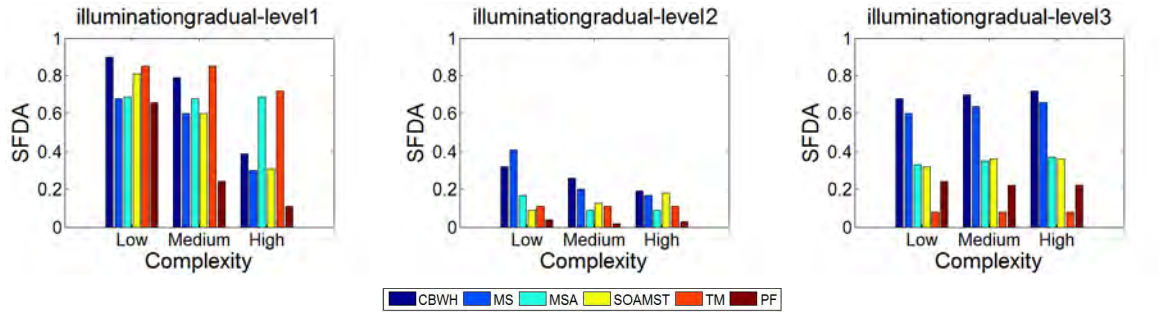


Figure 5.9: SFDA values of selected tracking algorithms for Illumination Gradual sequences

5.2.2.1 Illumination gradual

Comparing the plots in Figure 5.9 the most characteristic information is the extreme decrease of success in Level 2 sequences for certain algorithms, where results were far worse than expected.

In Level 1 (synthetic sequences) the best overall results are provided by TM. This is explained due to the background of the test sequences (uniform black color). Hence, illumination changes reduce the values resulting from the matching procedure but the target position still got the highest value due to the high difference between the target and background colors. CBWH and SOAMST also had good values for the low complexity level but decrease at a much higher rate. MSA provided good and constant results demonstrating that in controlled conditions and assuming that there are not abrupt changes in illumination, the algorithm does not lose the target and therefore delivers good results. Lastly, MS decreased fast, with low results for high complexity sequences, and PF does not perform correctly for medium complexity sequences.

In Level 2 (lab sequences) the most noticeable change is the extreme decrease in performance for MSA and SOAMST. CBWH, MS and TM had the best results, while PF delivered a medium performance. The low results for MSA and SOAMST can be explained due to the increase of the global illumination made that both algorithms lost the target in the beginning of the sequence without finding it again, therefore delivering poor results. Lastly, CBWH got the most stable results when changing the complexity of the sequences.

Level 3 (real sequences) seems to group algorithms clearly in three groups: CBWH, and MS provided high performance, MSA, SOAMST and PF obtained medium performance and finally TM provides extremely poor results, due to the fact that real complex sequences provide targets with higher noise and lower resolution, therefore making it difficult to create an accurate template to be followed. An example of the execution of the algorithms for the Level 3 sequence "l3_faces_illuminationglobal_medium" can be found in Figure 5.10. The only algorithm with problems following the target in this sequence was TM, as expected in algorithms with illumination changes: since the template created in the first frame is static, once the target starts changing its color (and therefore, the histogram), TM loses the target and can not find it again.

Figure 5.10: Example of gradual illumination changes with a Level 3 sequence. Frames: 1, 20, 40, 60. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

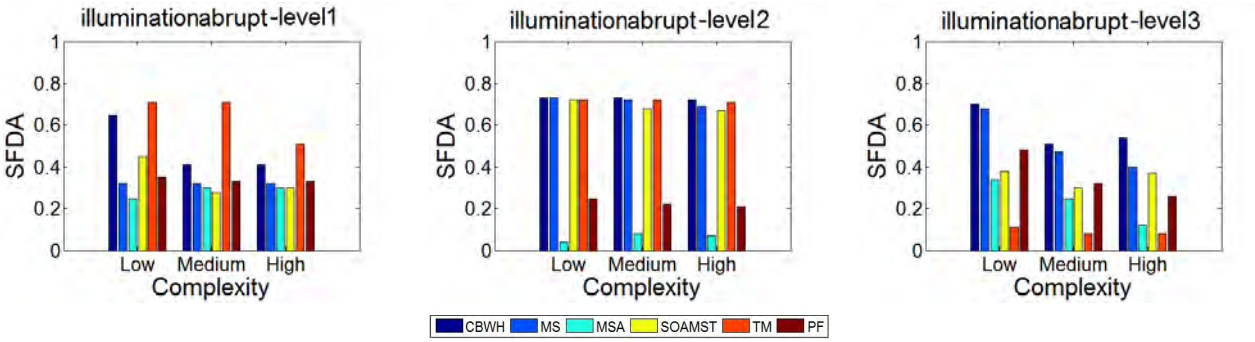


Figure 5.11: SFDA values of selected tracking algorithms for Illumination Abrupt sequences

5.2.2.2 Illumination Abrupt

The results for abrupt illumination changes can be seen in Figure 5.11. A noticeable trend is observed for most algorithms that seem to have problems with synthetic sequences (Level 1) but not real simple sequences (Level 2).

Results from Level 1 (synthetic sequences) showed that TM followed by CBWH provided the better performance. However, a decrease was observed for complex sequences. MS, MSA, SOAMST and PF reduced their performance as compared to TM and CBWH. However, they maintained similar accuracy for the different complexity levels.

Level 2 (lab sequences) results were the most peculiar, since a very high success rate was achieved not only with TM (which has the best performance in the previous level) but also CBWH, MS and SOAMST. PF presented the low performance, and finally MSA provided the worst results, demonstrating the non-adequacy of blind update scheme. Overall, the results for this level are very constant no matter the complexity degree.

In Level 3 (real sequences) best and most consistent results are obtained for CBWH and MS. As a general trend, there is a performance decrease as the complexity increases. The performance of the other algorithms was medium for MSA and PF and low for SOAMST and TM. An example of the execution of the algorithms for the Level 3 sequence "dtneu_nebel_taxi" can be found in Figure 5.12. In this example, both MSA and TM have problems following the target due to the actualization of the template, whereas the rest of the algorithms follow the target with no problems.

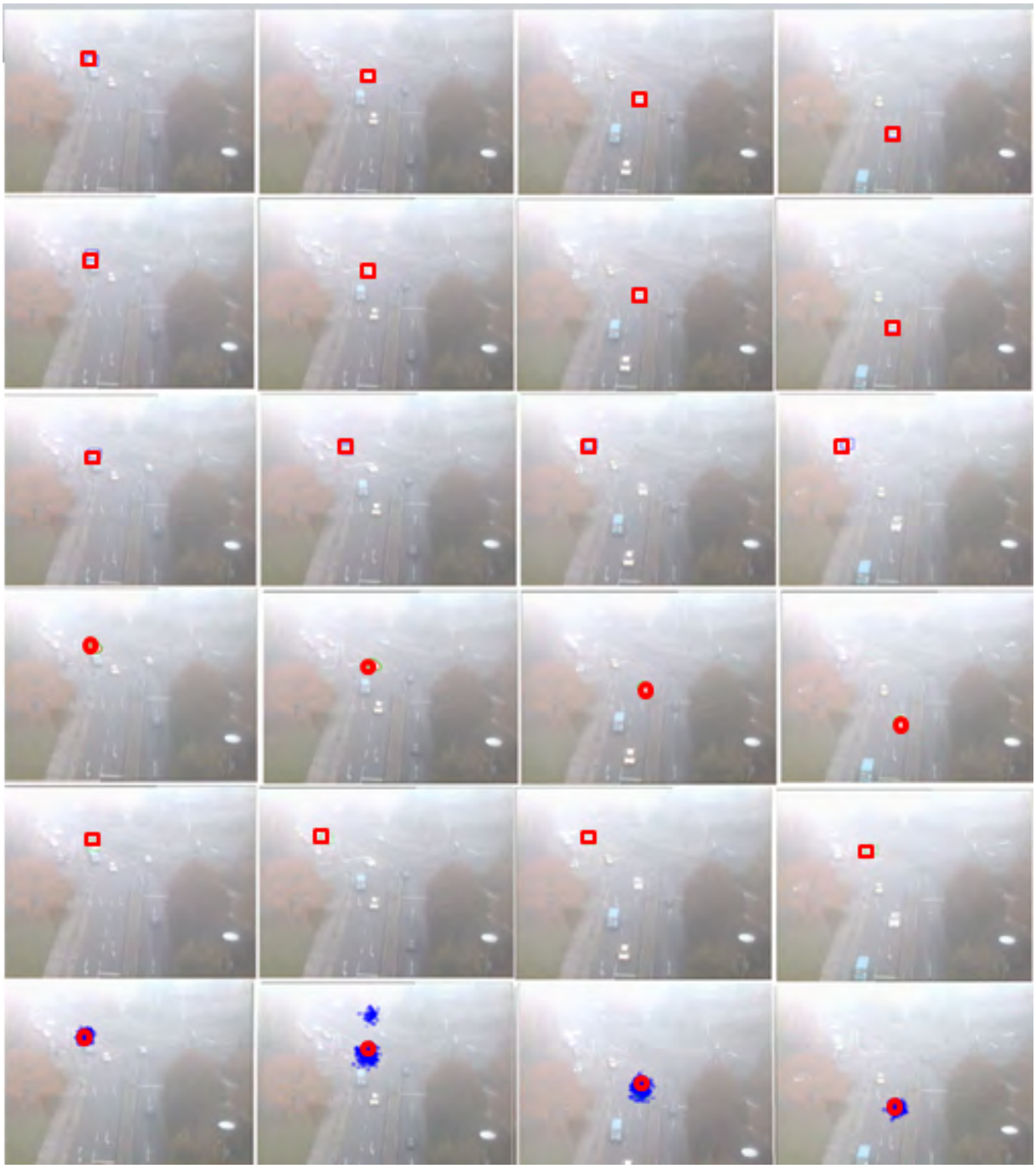


Figure 5.12: Example of abrupt illumination changes with Level 3 sequence. Frames: 1, 50, 100, 150. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

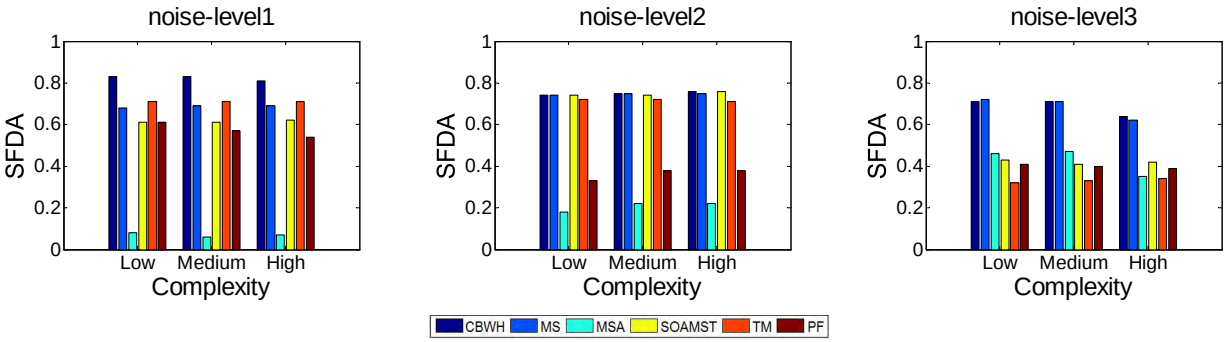


Figure 5.13: SFDA values of selected tracking algorithms for Noise sequences

5.2.2.3 Noise

Noise results shown in Figure 5.13 reveal that this is one of the issues where the algorithms provided good results in all levels, that is, being robust to sequences with noise.

In Level 1, CBWH had high performance, as well as MS, SOAMST, TM and PF (with slightly less good results). MSA presented very poor results. All algorithms delivered a constant performance without much changes regarding the complexity of the sequences.

In Level 2 all algorithms obtained the same accuracy independently of the degree of difficulty. CBWH, MS, SOAMST and TM all provided approximately the same results (very high SFDA values), with PF delivering medium results and lastly MSA providing poor results (although slightly better than the ones obtained for Level 1).

In Level 3 CBWH and MS again provided very good performance, with PF providing close result to them. Surprisingly, MSA delivered medium values (much better than in previous levels), similar to the ones obtained with SOAMST, which indicates that these algorithms tolerate better the existence of real noise (such as snow falling, or image noise) than the one added artificially with software. TM in this level got a lower performance compared to previous levels. An example of the execution of the algorithms for the Level 3 sequence "dtneu_schnee_redcar" can be found in Figure 5.14. Once again, MSA shows that it is not capable of dealing with many of the issues analyzed, whereas the rest of the algorithms can cope with slight changes due to noise in the sequence.

Figure 5.14: Example of noise with a Level 3 sequence. Frames: 1, 50, 100, 150. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

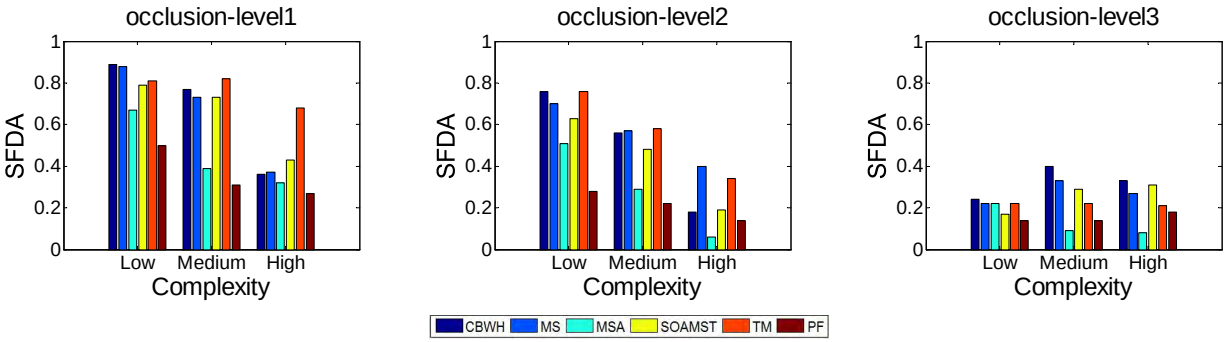


Figure 5.15: SFDA values of selected tracking algorithms for Occlusion sequences

5.2.2.4 Occlusion

The occlusion results presented in Figure 5.15 show interesting information for the first two levels. As seen in the sub figures, the increasing complexity level means a decrease in the success rate (SFDA) as expected. The most noticeable aspect is the general decrease of success for the third level.

In Level 1 (synthetic sequences) it is noted that CBWH, MS, MSA and SOAMST algorithms provided good results for the first two complexity levels and then experienced a big accuracy decrease for high complexity sequences. SOAMST, on the other hand, had approximately the same good results independently of the complexity and the same happened with PF, which had medium accuracy. This is due to the fact that even though the algorithm finds the object once it is visible again, the area is significantly smaller, and therefore, the accuracy decreases. In theory, if the sequences were longer after the object appears in the scene again, the size would increase and so would the accuracy.

In Level 2 (lab sequences) the same pattern is observed, with good results for the first two sub levels and a noticeable decrease for the last one in all algorithms. In this case, MS and SOAMST are the best options, followed by CBWH.

Level 3 (real sequences) showed a general decrease of performance for all algorithms. In this scenario all algorithms performed approximately the same with exception of MSA and PF with worse results. This noticeable decrease may be due to the higher complexity of the sequences in this level, therefore reducing the success rate for all algorithms. The obtained results demonstrate that partial (low complexity) and total (high complexity) occlusions are still an open issue as all the algorithms obtained low performance. An example of the execution of the algorithms for the Level 3 sequence "visor5_man_head" can be found in Figure 5.16. In this complex sequence with a total occlusion, all algorithms lose the target (frames 35 and 160) with the exception of PF that does not mistake the target with the occluding object (as seen in frame 35 for the other algorithms).

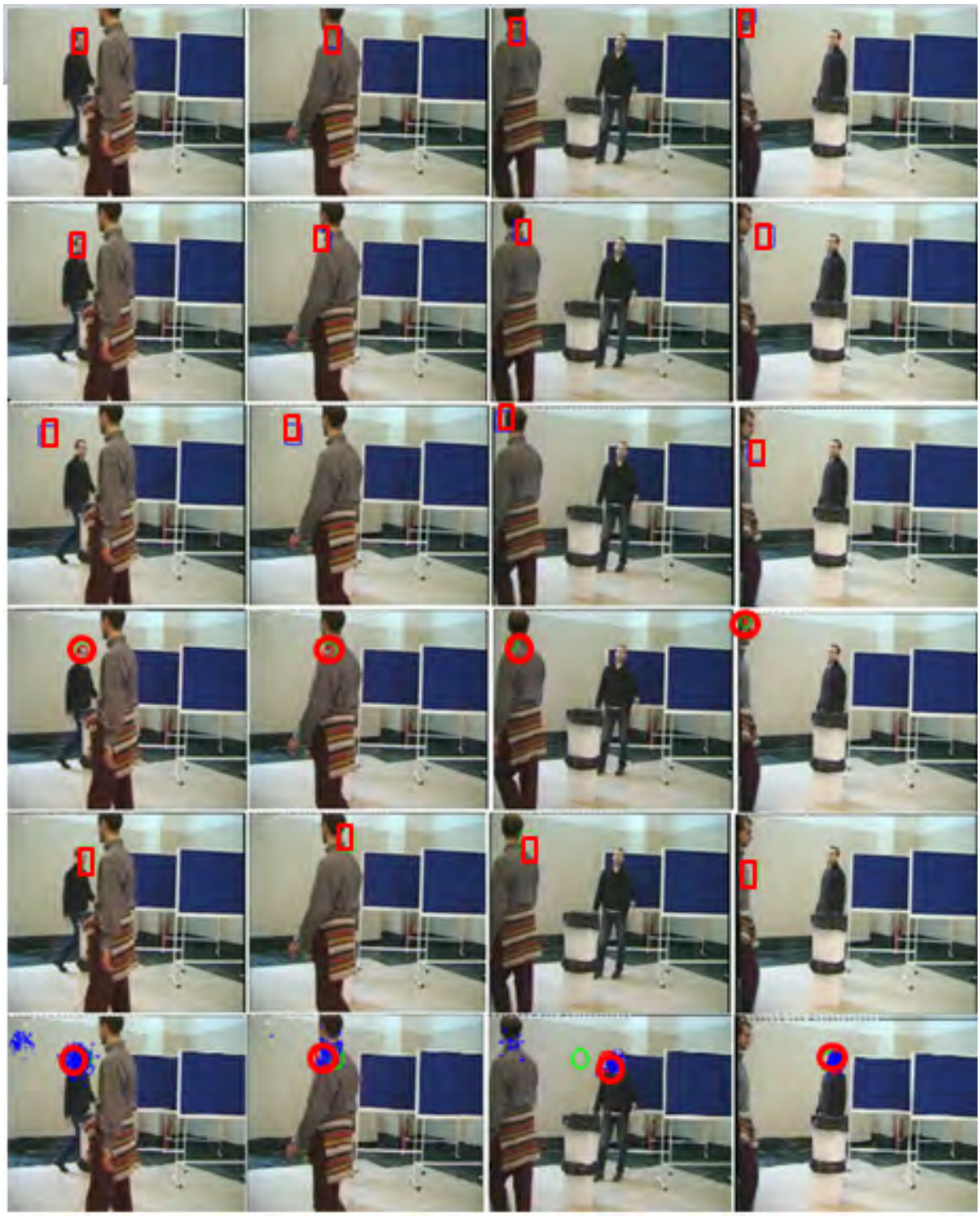


Figure 5.16: Example of occlusion with a Level 3 sequence. Frames: 15, 20, 35, 140. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

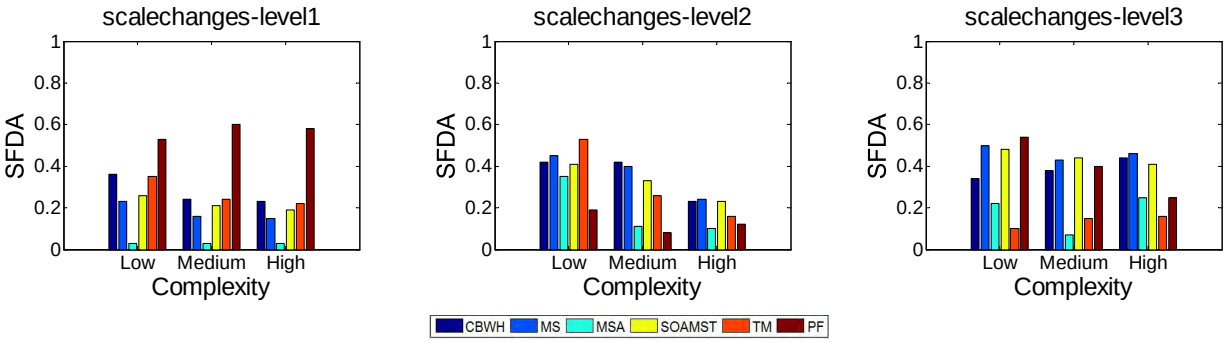


Figure 5.17: SFDA values of selected tracking algorithms for Scale Changes sequences

5.2.2.5 Scale changes

Figure 5.17 depicts the results for sequences where the target changes its size during the sequence. The overall performance for this issue had lower success rates than the other ones. Also, no clear pattern is observed.

In Level 1 (synthetic sequences) results were specially low no matter the degree of difficulty. The best algorithm was PF, with very high accuracy values, and with a big difference over the second best, CBWH, with SFDA results lower than 0.4. CBWH, MS, SOAMST and TM all obtained similar results and finally MSA delivered extremely low success rates.

In Level 2 (lab sequences) the overall SFDA was raised especially in the first complexity level, where results were medium (at best). The rest of the complexity levels presented decreasing values as the complexity increased obtaining low performance.

Finally, in Level 3 (real sequences), PF had a good performance as well as MS, SOAMST and CBWH. The fact that algorithms that are not prepared to deal with scale changes (such as MS and CBWH) obtained good results is due to the fact that most of the scale changes sequences show a decrease in the size (i.e., the target moves further away from the camera). Since MS and CBWH have a bigger search area than the real target, it is easy to find the target, although the accuracy lowers since the search area is obviously too big as the frames pass. Neither MSA nor TM performed well. An example of the execution of the algorithms for the Level 3 sequence "PETS2009_S2_L1_girl" can be found in Figure 5.18. In this example we can see once again that MSA loses the target once more information from the background starts appearing in the template created in each frame. TM also loses momentarily the target (frame 20), although it is find once again (frame 35)

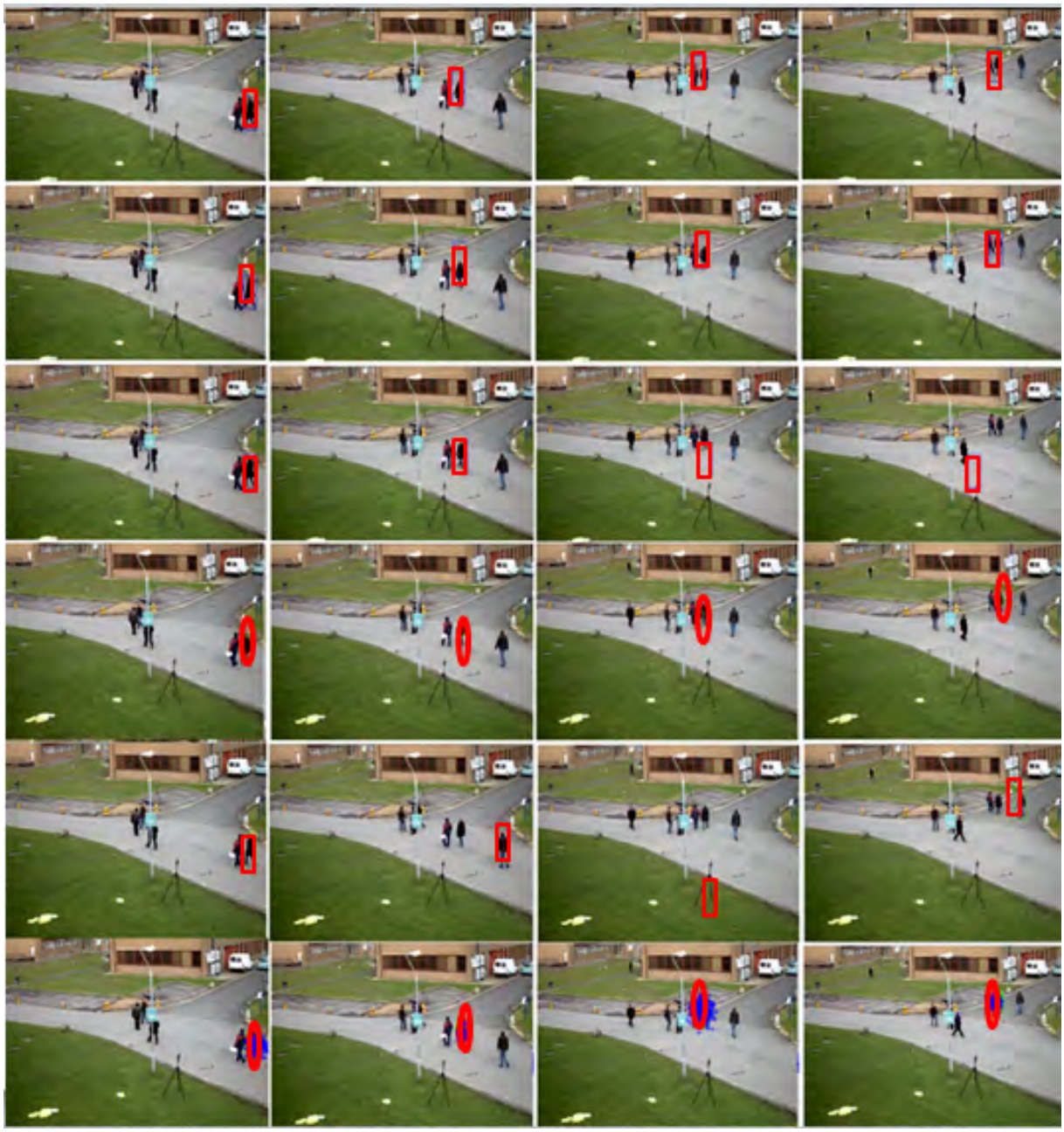


Figure 5.18: Example of scale changes with a Level 3 sequence. Frames: 1, 10, 20, 35. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

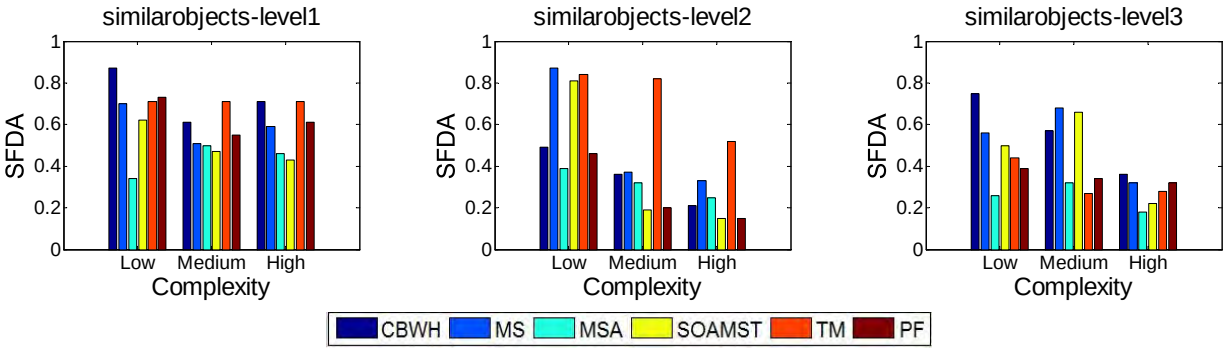


Figure 5.19: SFDA values of selected tracking algorithms for Similar Objects sequences

5.2.2.6 Similar objects

The results for the sequences with similar objects are shown in Figure 5.19. Again in this case there are no clear patterns for the three levels.

Level 1 (synthetic sequences) presented high performance for CBWH, MS, SOAMST and PF. TM and MSA both had medium-good results. The general increase in the accuracy for this issue may be due to the fact that sequences were less complex in general.

In Level 2 (lab sequences), TM showed high performance, especially as the sequences get more complex. Since TM uses the target to create a template, as far as that template is properly created, the algorithm will be robust against similar objects. With the exception of the peak in the low complexity level for MS and SOAMST, the other algorithms behaved as expected, decreasing the accuracy as the complexity increases.

In Level 3 (real sequences), CBWH and MS showed the best and most consistent results, while SOAMST, TM, PF and MSA obtained medium to good results, which decreased as the complexity increases. An example of the execution of the algorithms for the Level 3 sequence "PETS2009_S2_L2_jeans" can be found in Figure 5.20. In this example MSA shows the same behavior once again. TM does lose the target in frame 35.

5.2.2.7 Real Sequences Comparison

In this section we group the results of the real sequences by type of target (cars, faces and people) instead of their complexity degree.

Figure 5.21 shows results for Level 3 sequences, grouped by type of target. Best results are obtained for Faces, followed closely by Cars, and coming in third position, People, although results are more or less constant, without big changes in each algorithm. The exception is MSA, which performs much better for Cars than for other types of target. TM, on the other hand, follows better People sequences. In general, even though some algorithms behave better depending on the target, the best options overall are CBWH and MS.

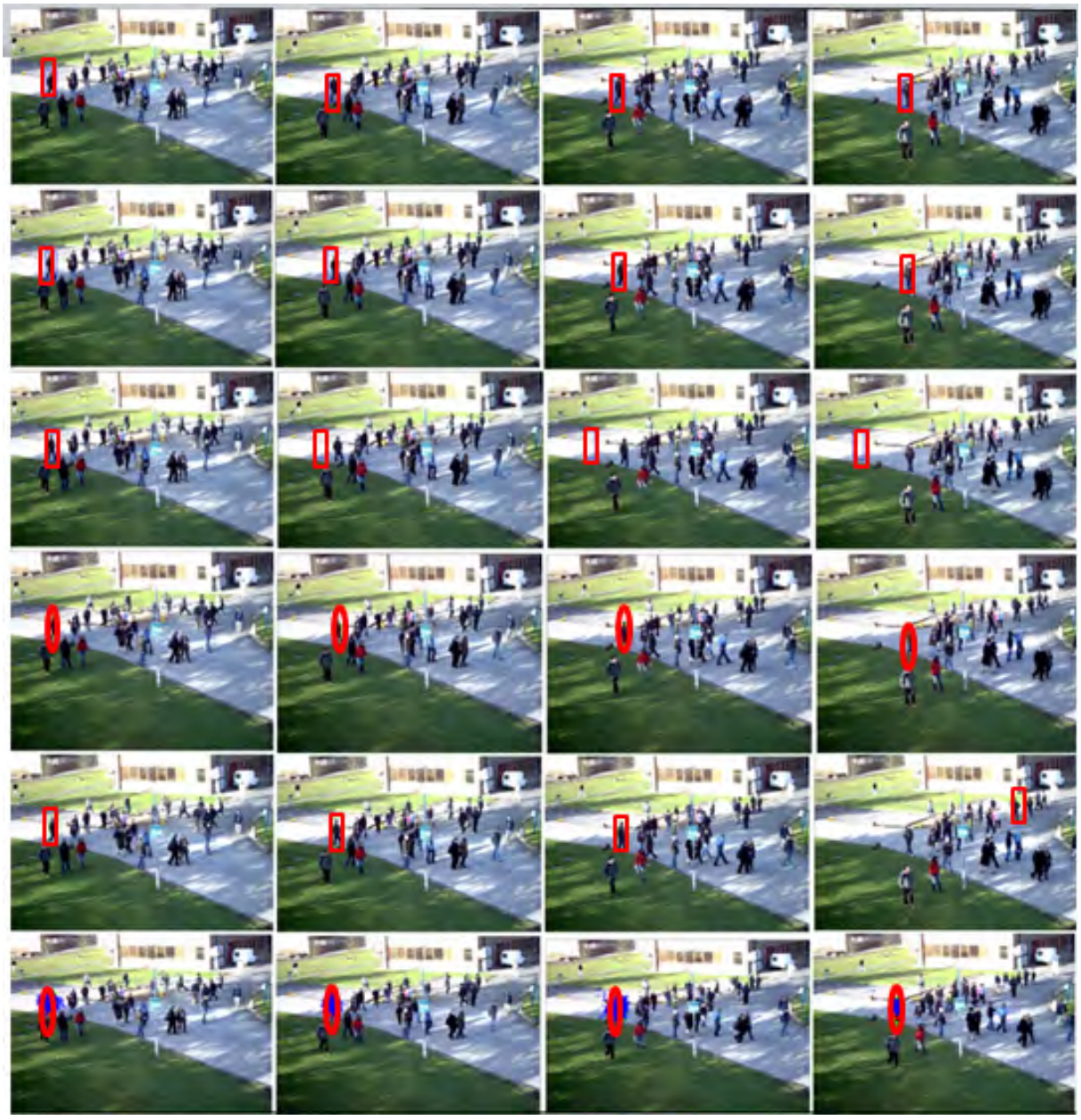


Figure 5.20: Example of similar objects with Level 3 sequences. Frames 1, 10, 20, 35. Algorithms (from top to bottom rows): CBWH, MS, MSA, SOAMST, TM, PF.

Figure 5.22 shows results for Level 4 sequences. In general, CBWH seems to have the best results, and there is a clear increase in the success rate for Faces sequences, where the target is easy to annotate (i.e., the target model contains a low amount of background data) and only a few different color regions are used to represent the target. Other than that there is not a clear pattern that allows to point the exact level of accuracy for each algorithm. It is very difficult to

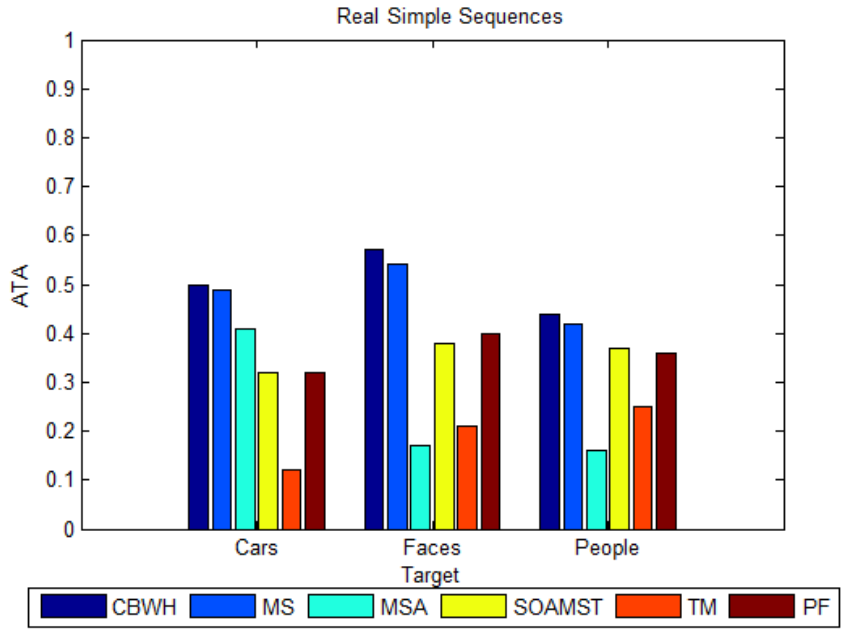


Figure 5.21: SFDA values for Real Simple sequences

accurately determine the complexity of each sequence, and therefore there is a possibility that Cars sequences have a higher degree of complexity than Faces or even People sequences.

General performance for sequences with several issues decreases in comparison with sequences with one isolated issue, as expected. The Faces sequences had better performance, which may be due to the lower complexity of the sequences or even the fact that those sequences are more easily annotated.

5.2.2.8 Conclusions

Once the results from all algorithms have been analyzed, the overall idea is that CBWH and MS are the most consistent algorithms. When dealing with real life scenarios, and according to the results from this dataset, those are the best options when presented with an unknown issue or even a combination of issues. Mean results for the selected algorithms and issues can be seen in Figure 5.23. At the right of the figure, the mean results for each algorithm are depicted: this allows to see at a first glance which algorithms perform better for the whole dataset. Results for PF show that the algorithm delivered the worst results, and after a careful study, this may be due to the fact that it is very difficult to accurately set the correct values for the parameters when the dataset includes very different targets. Another option is the fact that maybe the selected metrics are not appropriate for this kind of tracking. As mentioned in section 5.1, an SFDA of 0.6 means that the tracker works almost perfectly, while for success rates of around

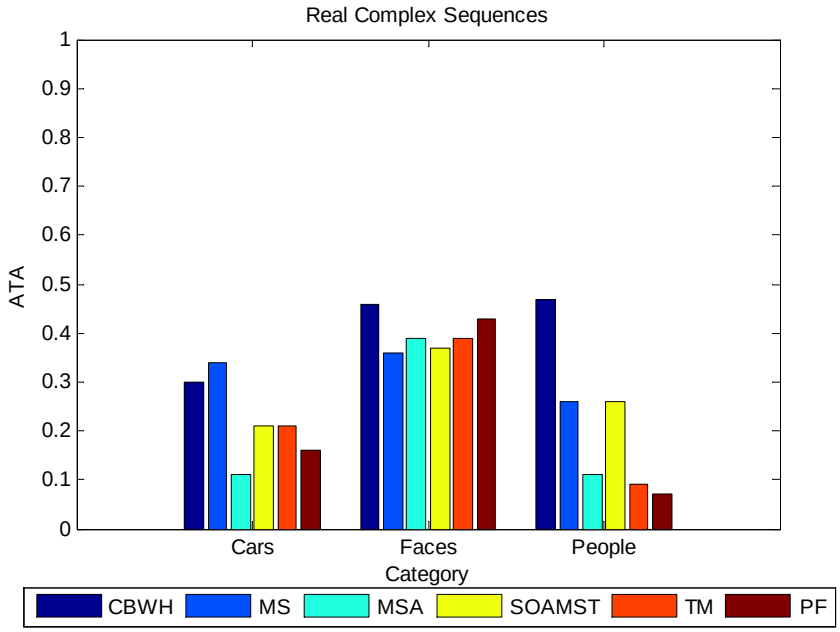


Figure 5.22: SFDA values for Real Complex sequences

50% the SFDA goes down to 0.2-0.3. This means that results in Figure 5.6 show success rates of more than 50% of track completeness in most cases.

One conclusion obtained was the fact that it is not a simple task to classify the real sequences by their complexity. Some sequences which appeared to be easy delivered in fact poorer results than others in appearance more complex.

Moreover, some examples are provided in Figure 5.24 for the algorithms executed on a sequence that contains a mixture of the following issues: similar objects in scene, occlusions and scale changes. Frame 235 shows the issue Similar Objects when a folder the same color as the target appears on the scene and moves close to it. In this case, only PF and TM are capable of correctly identifying the target: the deterministic algorithms (color-based) are not able to follow the target. Frame 420 show an Occlusion when the target changes its position and a slight Similar Object in the background. In this case, TM mistakes that similar object for the target (it is the closest to the template in the image). SOAMST has completely lost the target and stops tracking. Frame 500 (the last one) shows that all algorithms except SOAMST are capable of recovering after losing the target. This accounts for the low success rate of this algorithm.

Overall, the only algorithm that does not lose the target is PF, but the target area is much bigger than the real target. Therefore, since the result is not accurate, the SFDA metric will be lower and therefore provide poorer results. As a conclusion, PF is capable of tracking the object, but not correctly pointing the area it occupies.

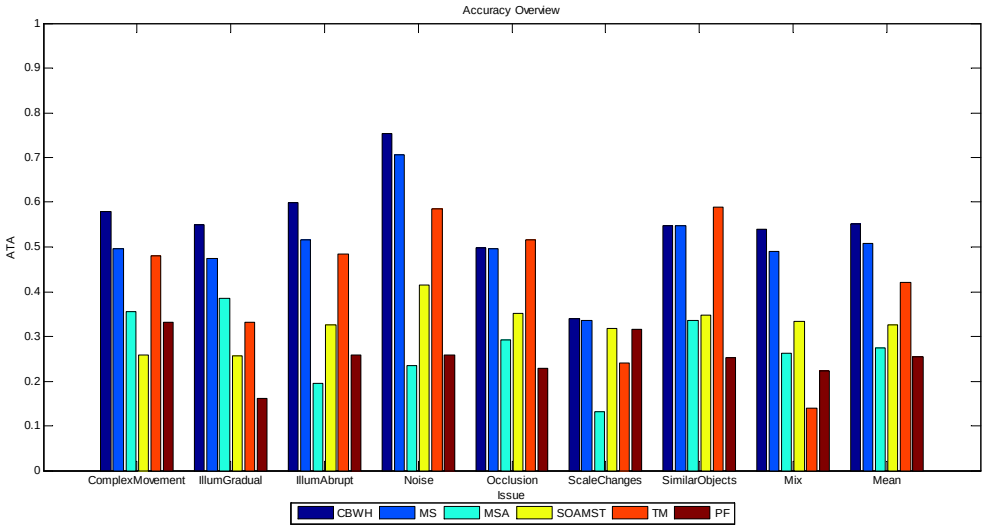


Figure 5.23: SFDA values overview for all sequences grouped by issue, including a mean value for all sequences together.

Issue	**SFDA**
Complex Movement	0.4685
Illumination Gradual	0.4229
Illumination Abrupt	0.3912
Noise	0.5263
Occlusion	0.3914
Scale Changes	0.2878
Similar Objects	0.4961
Mix	0.2957

Table 5.6: SFDA values for all sequences grouped by issue, including a mean value for all sequences together

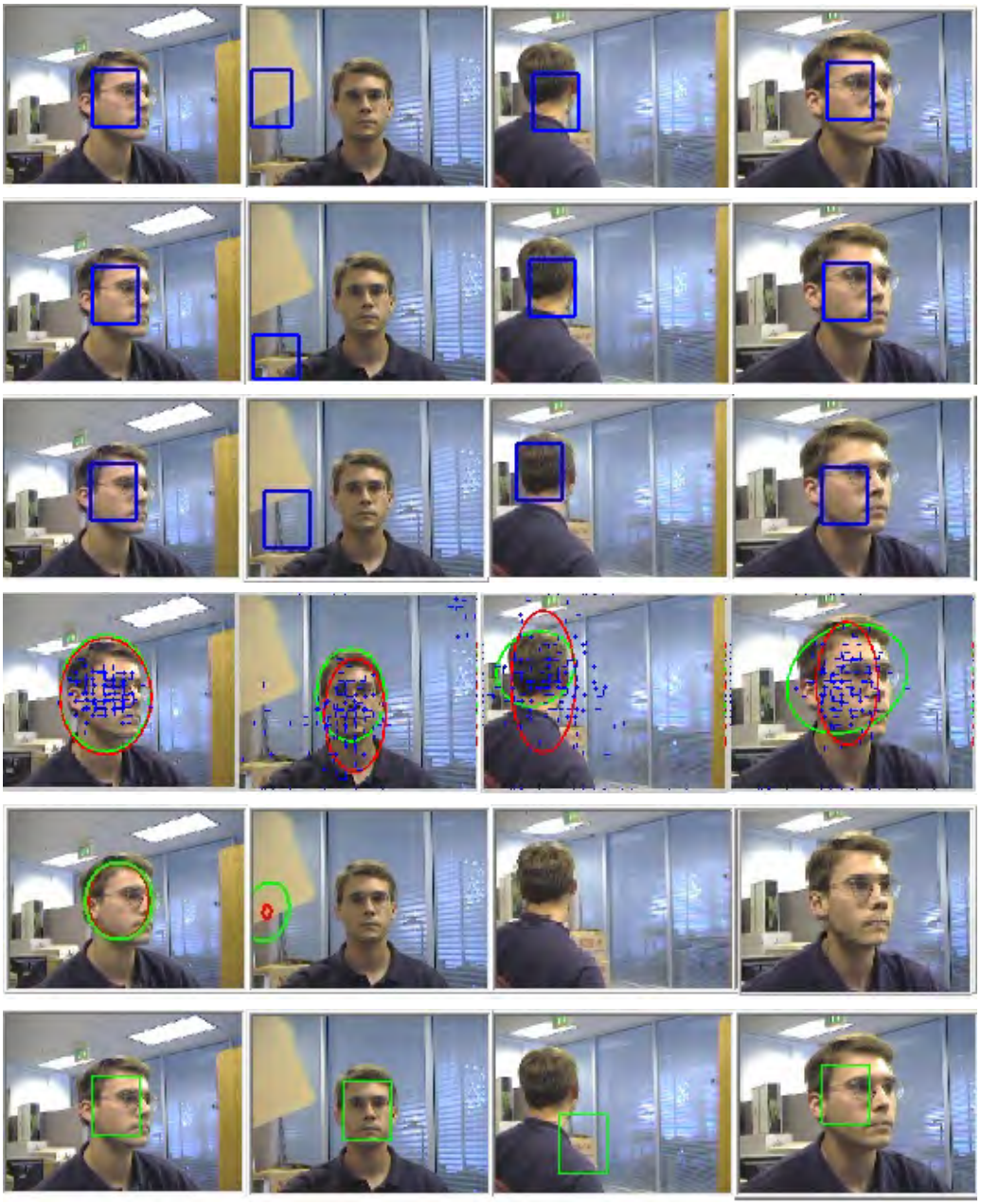


Figure 5.24: Example for Real Complex Sequence: "HEADTRACK_seq_sb". Frames: 1, 235, 420, 500 for each algorithm (from top to bottom rows): CBWH, MS, MSA, PF, SOAMST, TM.

Algorithm	Run 1	Run 2	Run 3	Run 4	Run 5	Run 6	Run 7	Run 8	Run 9	Run 10	Variance
CBWH	.539	.539	.539	.539	.539	.539	.539	.539	.539	.539	0
MS	.478	.478	.478	.478	.478	.478	.478	.478	.478	.478	0
MSA	.267	.267	.267	.267	.267	.267	.267	.267	.267	.267	0
SOAMST	.356	.356	.356	.356	.356	.356	.356	.356	.356	.356	0
TM	.386	.386	.386	.386	.386	.386	.386	.386	.386	.386	0
PF	.408	.410	.415	.411	.410	.415	.412	.411	.414	.411	5.417×10^{-6}

Table 5.7: Stability results for 10 runs for each algorithm

	CBWH	MS	MSA	PF	SOAMST	TM
Max	1.338	0.523	1.453	0.170	2.451	1.128
Min	0.045	0.030	0.045	0.003	0.313	0.023
Mean	0.333	0.139	0.340	0.050	0.589	0.124

Table 5.8: Comparative execution time per pixel (ms/pixel)

5.2.3 Stability

This experiment evaluates the variation of the generated tracking data when the is algorithm is run several times on the same sequence. In this case, the dataset was analyzed five times with each tracking algorithm and the mean value of SFDA was stored.

The obtained results are summarized in Table 5.7. As it can be observed, there are no differences between the runs for CBWH, MS, MSA, SOAMST and TM. This agrees with the definition of deterministic trackers, where results are non dependent of the runs without involving any kind of probabilistic (or random) processes. On the other hand, the PF did have different results with different runs. This is due to the fact that the algorithm is based on a probabilistic process where results always vary slightly. As seen in the results, this variation is minor and the range of values is compact, therefore, based on the SFDA metric there is no possibility of mistaking this algorithm's results with any other.

5.2.4 Efficiency

This experiment provides an insight on how time-consuming algorithms are. Hence, we measure the processing time for the application of the tracking algorithm to each sequence of the dataset. Then, we average this processing time by considering the size of the target (in pixels) and the number of frames of the sequences.For this task, all algorithm implementations were done in Matlab.

Results are summarized in Table 5.8 as the maximum, minimum and mean values per pixel are presented. The slower algorithm is clearly SOAMST, followed by MSA, whereas PF proved to be the fastest one. The difference between CBWH and MSA versus MS is due to the fact that the window size of the first two algorithms is much bigger than the one for MS, therefore increasing the processing time of the algorithm.

Chapter 6

Conclusions and Future Work

6.1 Summary

In this document we have presented a new evaluation protocol for video object tracking allowing the evaluation of different scenarios and providing a framework to compare the performance of the tracking algorithms.

First of all, an extensive research regarding the related work was done. It was required to understand the video tracking process, as well as to study existing metrics used and datasets for video tracking evaluation. This in-depth study was performed in chapter 2.

Afterward, a complete research of several algorithms was done. Once a clear understanding of (color-based) algorithms was obtained, a selection of the most relevant ones was performed. The main idea was to employ algorithms covering several approaches: including background information, scale and orientation changes, and deterministic and probabilistic approaches. All of them have in common the used feature: color. The information regarding the selected algorithms can be found in chapter 3.

After algorithm selection and implementation, it was necessary to develop a protocol for their evaluation. Different aspects were defined to study their optimum parameters, robustness to initialization error, accuracy, stability and efficiency. For the accuracy evaluation, the existing amount of metrics is overwhelming, although after a careful study, it became obvious that most of the metrics provided redundant information, or did not perform as well as expected due to details that were omitted when defining such metrics. Therefore, and analysis of some of the most relevant metrics was performed, and once it was clear which metrics provided good results, a selection of those metrics was included in the evaluation protocol in order to evaluate the sequences. Once the algorithms and metrics were ready, the necessity to create a complete dataset arose. When designing this dataset, the main goal was to cover as many scenarios, issues and situations as possible, and after an extensive study of existing datasets, we selected, created and annotated a total of 122 sequences. All this work is detailed in chapter 4.

Finally, the protocol was tested for evaluating the selected algorithms. The first goal was to test how the algorithms operate when certain issues appeared in the scenes as well as when other problems (such as bad initialization) happened. This step allowed to determine which algorithm performed better and provided the best results not only in each separate scenario but as a whole also. The second goal was to determine other details regarding how time-consuming each algorithm was, how different parameters affected the behavior of the algorithms or how different approaches (probabilistic vs. deterministic) affected the performance. All the information regarding the experimental results obtained when the proposed protocol was intensively tested can be found in chapter 5.

6.2 Final conclusions

After discussing the results of the selected algorithms in the previous sections, the best algorithm is CBWH as it had the highest mean success rate for the whole dataset. Moreover, it also runs faster than most of the compared algorithms, which is crucial when dealing with longer sequences. MS also obtained similar performance to that of CBWH which required a slightly additional computational time. SOAMST is definitely the worst algorithm, with low performance, erratic behavior and high processing time. However, this algorithm was the one with best behavior in sequences where the annotation included some sort of error (either in position, size, or a combination of both), due to the non-static search area which allows the algorithm to adjust the search area in order to accurately find the target. MSA, the adaptive version of MS, does not deliver good results since the update of the model makes it harder for the algorithm to find the target due to the inclusion of undesirable traces. Lastly, PF performed worse than expected which, as previously stated, may be due to the inappropriate metrics employed or the fact that it was not possible to accurately configure all parameters for such a disparate dataset. Note that all conclusions, results and comparisons are valid for this specific dataset.

In the next sections, we discuss the obtained results with respect to different characteristics of the selected algorithms.

6.2.1 Fixed template

In this case we have two algorithms: MS and TM. Both are simple methods, but whereas TM searchs for a certain fixed image by comparing pixels, MS uses histograms (color histograms in this implementation) in order to find the better match. This improvement is clearly shown in the results: MS has a better behavior overall. The disadvantage is that MS is twice as time consuming as TM is. In some cases such as complex movement, abrupt illumination changes and similar objects scenarios both algorithms show a similar behavior. For occlusion scenarios TM is a better option due to the fact that the search box is the whole image, and therefore, once the

target becomes visible again (the occlusion ends) TM finds it again faster than other algorithms (such as MS), but MS performs better in gradual illumination changes, noise, scale changes and most importantly, complex sequences with a number of issues due to the obvious limitations of using a fixed template (TM) instead of a color histogram (MS).

6.2.2 Adaptative template

In this case, a modification of the MS was implemented so that the algorithm took into account information from the last frame: in this case, instead of creating a template from the first frame of the target, the model was updated in each frame. Even though this might seem an advantage, in reality the algorithm performed much poorly since errors accumulated throughout the tracking process (drift). Therefore, a blind adaptative scheme model is not recommended.

6.2.3 Background information

Background information is used in CBWH algorithm (which is an update of the MS algorithm), delivering the best results overall for the dataset. The only disadvantage is the fact that since more steps are performed for each frame, the algorithm is slower. This may be solved by decreasing the target window size (although a slight decrease in the SFDA would be expected). Overall, the inclusion of background information proved to be a great addition to the basic MS algorithm.

6.2.4 Scale information

For this case we have two algorithms: SOAMST and PF. Both deal with the scale changes problem, one from the deterministic point of view (SOAMST) while the other uses a probabilistic approach (PF). In this case, both algorithms are more or less tied up with regards to their success rate: in some cases SOAMST presents a better behavior while in other issues it is PF the one that delivers better results. In sequences with a combination of different issues PF is clearly the best one. Since this algorithms are supposed to deal with scale changes, it is important to note that the behavior in that issue is not good: not only it is worse that other algorithms such as CBWH, but also it is worse than in other issues, for example similar objects. As a final note, PF is the fastest algorithm whereas SOAMST proves to be a extremely slow one: it takes more than 10 times longer than PF. Algorithms that do not include scale changes adaptation perform well with this dataset. A possible reason is the fact that if the scale change consists on the object moving further away (i.e. reducing its size) the algorithm has more possibilities of finding the target, therefore improving the overall accuracy (although the area is bigger than needed).

6.2.5 Deterministic vs. probabilistic approach

As seen in the experiments of chapter 5, there is a noticeable difference between the best algorithms (CBWH in most scenarios) and PF. Therefore, with this dataset and the implementations used for each algorithm, as well as using the metrics previously described, the deterministic approach provides much better results than the probabilistic one. The main advantage of PF is the slower execution time, while CBWH's best characteristic is the high SFDA values obtained.

6.3 Future Work

Results obtained in the previous section prove that some algorithms operate much better than others and are, therefore, a better option when designing a video tracking system. However, it is important to notice that those results can not be taken as a general rule, since there are some limitations of the protocol.

This document proposes the outline and implementation of an evaluation protocol, although several changes could be introduced in order to provide more extensive results.

- First of all, the feature used for all algorithms for modeling the target was color. This implementation has many advantages, but obviously, a more complex model could be used in order to study how that modification affects the behavior of the algorithms.

- Second, although the dataset was created as extensive as possible, it is important to note that only several issues were taken into account. Other sequences with strong appearance changes, or other interesting tracking problems could be created and included.

- Even though the whole system was done as automated as possible, it may be interesting to optimize the way the system works, especially if the dataset grows. This includes, but it is not limited to, the testing of all videos with each algorithm, the creation and evaluation of the annotation files for each test and the analysis of those files in order to determine the accuracy of each case.

- It would be interesting as well to check the time consuming more exhaustively, testing how different parameters values of the algorithms affect the efficiency.

- Moreover, it was observed that the definition of good tracking performance is still unclear as a spatial overlap of 50% (between target estimation and gound-truth annotation) might be valid for some applications (whereas not for others). Hence, new metrics without dependency on the application should be investigated (such as detailed in [7])

Bibliography

[1] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, us ed edition, August 2002. 1, 129

[2] E. Maggio and A. Cavallaro. *Video tracking: theory and practice*. Wiley, 2011. 1, 3, 8, 9, 10, 11, 13, 14, 60, 129, 130, 131

[3] C. Mcintosh and G. Hamarneh. Human limb delineation and joint position recovery using localized boundary models. In *IEEE Workshop on Motion and Video Computing*, page 1, 2007. 2, 130

[4] J. Xie, S. Khan, and M. Shah. Automatic tracking of escherichia coli bacteria. In *Medical Imaging Computing and Computer Assisted Intervention*, pages 824–832, 2008. 2, 130

[5] J. C. San Miguel, A. Cavallaro, and J. M. Martinez. Evaluation of online quality estimators for object tracking detection. In *IEEE Int. Conf. on Image Processing*, pages 825–828, 2010. 3

[6] F. Yin, D. Makris, and S. Velastin. Performance evaluation of object tracking algorithms. In *Int. Wokshop on Performance Evaluation of Tracking and Surveillance*, 2007. 4, 7, 15, 21, 22, 61, 131

[7] T. Nawaz and A. Cavallaro. Pft: A protocol for evaluating video trackers. In *IEEE Int. Conf. on Image Processing*, pages 2325–2328, 2011. 4, 19, 22, 23, 61, 92, 139

[8] V. Manohar, M. Boonstra, V. Korzhova, P. Soundararajan, D. Goldgof, R. Kasturi, S. Prasad, H. Raju, R. Bowers, and J. Garofolo. PETS vs. VACE evaluation programs: A comparative study. In *Int. Wokshop on Performance Evaluation of Tracking and Surveillance*, pages 1–6. 7, 16

[9] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:10, 2008. 7, 101

[10] F. Bashir and F. Porikli. Performance evaluation of object detection and tracking systems. In *Int. Wokshop on Performance Evaluation of Tracking and Surveillance*, pages 7–14. 7, 17, 18, 21

[11] C. Cannons. A review of visual tracking. Technical report, York Univesity, 2008. 8

[12] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006. 8, 9, 11, 13

[13] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Jorunal Computer Vision*, 60:63–86, October 2004. 8

[14] L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice Hall, January 2001. 8

[15] D. Conte, P. Foggia, G. Percannella, F. Tufano, and M. Vento. An experimental evaluation of foreground detection algorithms in real scenes. *EURASIP J. Adv. Signal Process*, 2010:7:1–7:10, February 2010. 8, 10

[16] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Int. Conf. on Machine learning*, pages 161–168, 2006. 8

[17] J. Park and Y. L. Murphey. Edge detection in grayscale, color, and range images. In *Wiley Encyclopedia of Computer Science and Engineering*. 2008. 10

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 10

[19] H. Ba, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *In Eur. Conf. on Computer Vision*, pages 404–417, 2006. 10

[20] S. C. S. Cheung and C. Kamath. Robust background subtraction with foreground validation for urban traffic video. *EURASIP J. Appl. Signal Process.*, 2005:2330–2340, January 2005. 10

[21] I.R. Khan and F. Farbiz. A back projection scheme for accurate mean shift based tracking. In *IEEE Int. Conf. on Image Processing*, pages 33–36, 2010. 12, 14

[22] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960. 12

[23] N. Gordon, D. Salmon, and S. Smith. A novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEEE Proc. of Radar and Signal Processing*, pages 107–113, 1993. 12, 40

[24] Z. Kalar et al. Forward-backward error: Automatic detection of tracking failures. In *IEEE Int. Conf. on Pattern Recognition*, 2010. 13

[25] C. Kim and J. Hwang. Fast and automatic video object segmentation and tracking for content-based applications. *IEEE Trans. Circuits Syst. Video Techn.*, 12(2):122–129, 2002. 13

[26] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:747–757, August 2000. 14

[27] R. Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley Publishing, 2009. 14, 25, 26, 61

[28] Y. Zhong and A.K. Jain. Object tracking using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:544–549, 2000. 14

[29] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Internation Journal of Computer Vision*, 1(4):321–331, 1988. 15

[30] A. Y. Xin, X. Li, and M. Shah. Object contour tracking using level sets. In *Asian Conference on Computer Vision*, pages 1–7, 2004. 15

[31] Y. Chen, Y. Rui, and T. S. Huang. Jpdaf based hmm or real-time contour tracking. *IEEE Computer Vision and Pattern Recognition*, 1:1–543, 2001. 15

[32] E. Maggio and A. Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, volume 2, pages 221 – 224, 18-23, 2005. 15

[33] A. Baumann, M. Boltz, J. Ebling, M. Koenig, H. S. Loos, M. Merkel, W. Niem, J. K. Warzelham, and J. Yu. A review and comparison of measures for automatic video surveillance systems. *EURASIP Journal on Image and Video Processing*, 2008:1–30, 2008. 15

[34] CLEAR 2007. http://www.clear-evaluation.org/?CLEAR, 2007. 16

[35] Computer for the human interaction loop. http://chil.server.de/. 16

[36] ETISEO. http://www-sop.inria.fr/orion/ETISEO/, 2007. 16, 107

[37] ETISEO@cvg. http://www.cvg.rdg.ac.uk/projects/etiseo/index.html, 2007. 16

[38] A.T. Nghiem, F. Bremond, M. Thonnat, and V. Valentin. Etiseo, performance evaluation for video surveillance systems. In *IEEE Int. Conf. on Advanced Video and Signal-based Surveillance*, pages 476–481, 2007. 17

[39] V. Manohar, P. Soundararajan, H. Raju, D. B. Goldgof, R. Kasturi, and J. S. Garofolo. Performance evaluation of object detection and tracking in video. In *Asian Conference on Computer Vision*, pages 151–161, 2006. 18, 19, 20, 61

[40] Z. Han, Q. Ye, and J. Jiao. Online feature evaluation for object tracking using kalman filter. In *IEEE Int. Conf. on Pattern Recognition*, pages 1–4, 2008. 19

[41] R. Kasturi, D Goldgof, P. Soundararajan, V. Manohar, M. Boonstra, and V. Korzhova. Performance evaluation protocol for face, person and vehicle detection and tracking in video analysis and content extraction. In *ARDA*, pages 151–161, 2006. 21

[42] D. M. Chu and A. W. M. Smeulders. Thirteen hard cases in visual tracking. In *IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, 2010. 21, 61

[43] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):319–336, 2009. 21, 101, 102, 103

[44] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(5), 2003. 25, 27, 29, 31, 32, 61

[45] J. Ning, L. Zhang, D. Zhang, and C. Wu. Robust mean shift tracking with corrected background-weighted histogram. *IET Computer Vision*, pages 1–27, 2012. 26, 31, 32, 34, 35, 61

[46] K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21:32–40, 1975. 27

[47] J. Ning, L. Zhang, D. Zhang, and C. Wu. Scale and orientation adaptive mean shift tracking. *IET-Computer Vision*, 6(1):52–61, 2012. 35, 36, 37, 38, 40, 61

[48] Gary R. Bradski. Computer Vision Face Tracking For Use in a Perceptual User Interface. *Intel Technology Journal*, 1998. 35

[49] E. Cuevas, D. Zaldivar, and R. Rojas. Particle filter in vision tracking. *Av Revolucion*, (August):1–11, 2005. 41, 42

[50] K. Nummiaro, E. Koller-Meier, and L. J. Van Gool. An adaptive color-based particle filter. *Image Vision Comput.*, 21(1):99–110, 2003. 41, 44

[51] Mit traffic data set. http://www.ee.cuhk.edu.hk/ xgwang/MITtraffic.html, 2009. 54, 57, 113, 115

[52] Institut fur algorithmen und kognitive systeme: Cars dataset. http://i21www.ira.uka.de/image-sequences/. 54, 115

[53] Trecvid 2009 dataset. http://trecvid.nist.gov/trecvid.data.html. 54, 117, 118

[54] S. Birchfield. Elliptical Head Tracking Using Intensity Gradients and Color Histograms. http://www.ces.clemson.edu/ stb/research/headtracker/. 54, 57, 117

[55] VISOR. http://www.openvisor.org/, 2007. 54, 111, 117

[56] PETS 2010. http://pets2010.net/, 2010. 54, 57, 107, 118

[57] i-lids web. http://scienceandresearch.homeoffice.gov.uk/hosdb/cctv-imaging-technology/video-based-detection-systems/i-lids/. 54, 118

[58] CAVIAR. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, 2007. 54, 110, 118

[59] PETS 2000 ftp. ftp://ftp.pets.rdg.ac.uk/pub/PETS2000, 2000. 54, 107, 118

[60] A. Perera, A. Hoogs, C. Srinivas, G. Brooksby, and W. Hu. Evaluation of algorithms for tracking multiple objects in video. pages 1–35, 2006. 101, 102, 103

[61] D. Roth, E. Koller-Meier, D. Rowe, T.B. Moeslund, and L. Van Gool. Event-based tracking evaluation metric. *Motion and Video Computing, IEEE Workshop on*, 0:1–8, 2008. 103

[62] D. Roth, E. Koller-Meier, and L. J. Van Gool. Multi-object tracking evaluated on sparse events. *Multimedia Tools Appl.*, 50(1):29–47, 2010. 104

[63] CANTATA. http://www.hitech-projects.com/euprojects/cantata, 2007. 105

[64] Surveillance performance evaluation initiative. http://www.eecs.qmul.ac.uk/ andrea/spevi.html, 2009. 105

[65] PETS 2001. http://www.cvg.cs.rdg.ac.uk/PETS2001, 2001. 107

[66] PETS 2006. http://www.cvg.rdg.ac.uk/PETS2006, 2006. 107

[67] PETS 2007. http://www.cvg.rdg.ac.uk/PETS2007, 2007. 107

[68] CAVIAR data. http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/, 2007. 110

[69] AVSS. http://www.eecs.qmul.ac.uk/ andrea/avss2007, 2007. 112

[70] Videos for head tracking. http://www.ces.clemson.edu/ stb/research/headtracker/seq/, 1998. 113

[71] X. Wang, X. Ma, and W. E. L. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31:539–555, March 2009. 113

Appendix A

Metrics for Multiple Object Tracking Evaluation

A.1 Multiple target evaluation

Although several approaches have been proposed for single object tracking evaluation, the multiple object tracking field still lacks a general agreement. The objective of multiple object tracking is to assign a unique ID to each object which stays constant throughout the sequence [9].

Therefore, with this objective in mind, a performance metric should be able to determine if the tracking algorithm is precise when determining the target's location and if the algorithm is able to follow the target as it moves through the scene even if it is temporally occluded (partially or completely).

In [43] a set of four metrics was developed, two for detection and two for tracking. These measures split the accuracy and the precision aspects of the system in two separate scores, and were the primary measures used to score algorithm performance in the CLEAR 2006 evaluation tasks.

A.1.1 Detection

A.1.1.1 Split Fraction

This frame-based metric provides information of whether the tracked targets are completely detected or if they are fragmented by the algorithm [60]. The ideal value is 1: a higher number means that single targets are being detected as multiple fragments.

$$Split\ Fraction = \frac{\#TP + \#split}{Total\ \#GT\ locations} \tag{A.1}$$

Merge Fraction This frame-based metric provides information of whether multiple tracked targets are completely detected or if they are merged together by the algorithm [60]. The ideal value is 1: a higher number means that multiple targets are being clustered together.

$$Merge\ Fraction = \frac{\#TP + \#merge}{Total\ \#GT\ locations} \tag{A.2}$$

Multiple Object Detection Accuracy It measures the missed detection and false positive counts. If m_t is the number of missed detections and fp_t is the number of false positives (in a frame t), the Multiple Object Detection Accuracy (MODA) [43] can be computed as:

$$MODA(t) = 1 - \frac{c_m(m_t) + c_f(fp_t)}{N_G^{(t)}} \tag{A.3}$$

where c_m and c_f are the cost functions of the missed detects and false positive penalties. The values for these weights can be changed based on the application.

Multiple Object Detection Precision It uses the spatial overlap info between ground-truth and system output as in section **??**.

$$OverlapRatio = \sum_{i=1}^{N_{mapped}^{(t)}} \frac{|G_i^{(t)} \cap D_i^{(t)}|}{|G_i^{(t)} \cup D_i^{(t)}|} \tag{A.4}$$

Therefore, for each frame (t), the Multiple Object Detection Precision[43] can be computed as:

$$MODP(t) = \frac{OverlapRatio}{N_{mapped}^{(t)}} \tag{A.5}$$

Multiple Object Count It measures the count accuracy over the whole sequence [43].

$$MOC = 1 - \frac{m + f_p}{N_G} \tag{A.6}$$

If the system is ideal, $MOC = 1$, meaning that there are no misses or false positives.

A.1.2 Tracking

A.1.2.1 Track Completeness Factor

This track-based metric provides measures how well a given object is detected after the association [60].

$$TCF = \frac{\sum_i \sum_{T_j \in A(G_i)} |O\left(T_j, G_i\right)|}{\sum_i |G_i|} \tag{A.7}$$

A.1.2.2 Track Fragmentation [60]

$$TF = \frac{\sum_i |A(G_i)|}{|\{G_i | A(G_i) \neq 0\}|} \tag{A.8}$$

A.1.2.3 Normalized Track Fragmentation [60]

The normalization increases the weight for longer tracks, to account for the fact that it is more difficult and important to maintain the identity of long tracks than short ones. A fragmentation score of n means that we have identified the target with n labels (tracks). A fragmentation of 1 is the ideal.

$$NTF = \frac{\sum_i |G_i| \cdot |A(G_i)|}{\sum_{i|A(G_i)\neq 0|} |G_i|} \tag{A.9}$$

A.1.2.4 Multiple Object Tracking Accuracy

The aim is to extract the accuracy of the system output track by computing the number of missed detects and false positives as well as the switches in the system output track for a given ground-truth annotation [43].

$$MOTA = 1 - \frac{\sum_{i=1}^{N_{frames}} (c_m(m_i) + c_f(fp_i) + log_e(id_{switches})}{\sum_{i=1}^{N_{frames}} N_G^i} \tag{A.10}$$

A.1.2.5 Multiple Object Tracking Precision

Calculates the spatiotemporal overlap between ground-truth annotations and system output [43].

$$MOTP = \frac{\sum_{i=1}^{N_{mapped}} OverlapRatio}{\sum_{j=1}^{N_{frames}} N_{mapped}^j} \tag{A.11}$$

where

$$OverlapRatio = \sum_{t=1}^{Nframes} \frac{|G_i^{(t)} \cap D_i^{(t)}|}{|G_i^{(t)} \cup D_i^{(t)}|} \tag{A.12}$$

A.2 Event-driven metrics

When a person views a scene, the way he conceptualizes the world is by means of events and objects, so a motivation to evaluate tracking performance on a higher level was originated. In [61], a new metric was proposed, capable of extracting different types of higher level events such as entering the scene, occlusion or picking-up a bag from the available data. This metric then focuses on the completeness of such event detection to do the evaluation of tracking data.

An event is the basic building block and it is defined as a limited action at a particular point in time. Some examples of events are entering/leaving a scene or a specific area (such a shop), starting/ending occlusion, starting/ending walking or running..

Advantages of using a event-based evaluation are (as established in [62]):

- The lengths of trajectories do not influence the metric making it independent of the frame rate and density of the ground-truth annotation.
- It enables the fast generation of ground-truth data as not every frame needs to be annotated in full detail, as long as the events can be reliably extracted from sparse annotation.
- Reuse of already available ground-truth data by automatic conversion into our novel event-based representation.
- Minimizing the human factor within the ground-truth data and its influence onto the metric by means of event-based evaluation on a higher level.
- Establishing a least common denominator to represent tracking data which is versatile to handle many different output formats.
- The metric directly helps to improve tracking algorithms by identifying: difficult trajectories, difficult scene locations, difficult situations and difficult event types.
- Easy integration into higher level event and object detection frameworks

Appendix B

Datasets for video object tracking

Nowadays there are several datasets available for video object tracking. Some are very complete whereas others are specific for a particular issue. In this section, we present the most interesting tracking related datasets for faces, people and cars.

B.1 CANTATA

The Content Aware Networked systems Towards Advanced and Tailored Assistance (CANTATA) project was developed during a 3 year period with the main goal of expanding the actual content-aware systems in the following aspects [63]: multimodal information fusion, scalable systems, applications flexible in multiple dimensions, guarantee of proper performance levels under all circumstances.

In the context of the European CANTATA project, partners involved in multi content analysis validation methods combined their efforts to create a webpage to share knowledge about datasets (sets, metadata, ground-truth, metrics...) for three different domains: surveillance, consumer applications and medical. Therefore, CANTATA contains a selection of the available datasets for video tracking up to 2008.

B.2 SPEVI

The Surveillance Performance EValuation Initiative (SPEVI) [64] is a set of links of publicly available datasets for researches. The videos can be used for testing and evaluating video tracking algorithms for surveillance-related applications. Two datasets are especially interesting regarding the tracking evaluation and they are described as follows.

Figure B.1: Frames 0, 100, 200 and 300 of the "motinas_emilio_webcam.avi" video of single faces dataset (SPEVI)

B.2.1 Single Face Dataset

- Description: this is a dataset for single person/face visual detection and tracking. The sequences include different illumination conditions and resolutions.
- Number of sequences: 5 sequences, 3018 frames.
- Format: individual JPEG images.
- Tracking ground-truth available: yes.

B.2.2 Multiple Faces Dataset

- Description: this is a dataset for multiple people/faces visual detection and tracking. The sequences (same scenario) contain 4 targets which repeatedly occlude each other while appearing and disappearing from the field of view of the camera.
- Number of sequences: 3 sequences, 2769 frames.
- Format: individual JPEG images.
- Tracking ground-truth available: yes.

Figure B.2: Frames from "motinas_multi_face_turning" and "motinas_multi_face_fast" of multiple faces dataset (SPEVI)

Figure B.3: Example of ETISEO images

B.3 ETISEO

As already introduced in 2.3.2, ETISEO [36] is a video understanding evaluation project that contains the following dataset.

- Description: these sequences contain indoor and outdoor scenes, corridors, streets, building entries, subway station... They also mix different types of sensors and complexity levels.
- Number of sequences: 86 sequences.
- Tracking ground-truth available: yes.

B.4 PETS

PETS [59, 65, 66, 67, 56] is the most extended database nowadays. As explained in 2.3.2, a new database is released each year since 2000, along with a different challenge proposed. With the

algorithms provided researchers can test or develop new algorithms. The best ones are presented in the conference held each year.

Since the amount of data is extensive and cover real situations, these databases are by far the most used and are almost considered a *de facto* standard. Despite this, it is important to say that the PETS databases are not ideal. One of its disadvantages is the fact that since PETS became a surveillance project, the challenges proposed are focused on high level applications of that field, leaving aside the tracking approach. Therefore, some important issues (such as illumination or target scale changes) are not considered.

B.4.1 PETS 2000

- Description: Outdoor people and vehicle tracking (single camera).
- Number of sequences: 1 set of training and test sequence.
  - Training sequence: 3672 frames.
  - Test sequence: 1452 frames.
- Formats: MJPEG movies and JPEG frames.
- Tracking ground-truth available: no.

B.4.2 PETS 2001

- Description: Outdoor people and vehicle tracking (two synchronized views; includes omnidirectional and moving camera). Challenging in terms of significant lighting variation, occlusion, scene activity and use of multi-view data.
- Number of sequences: 5 sets of training and test sequences
  - Training sequences: 1st) 3064 frames. 2nd) 2989 frames. 3rd) 5563 frames. 4th) 6789 frames. 5th) 2866 frames.
  - Test sequences: 1st) 2688 frames. 2nd) 2823 frames . 3rd) 5336 frames. 4th) 5010 frames. 5th) 2867 frames.
- Formats (for each set): MJPEG movies and JPEG frames.
- Tracking ground-truth available: no.

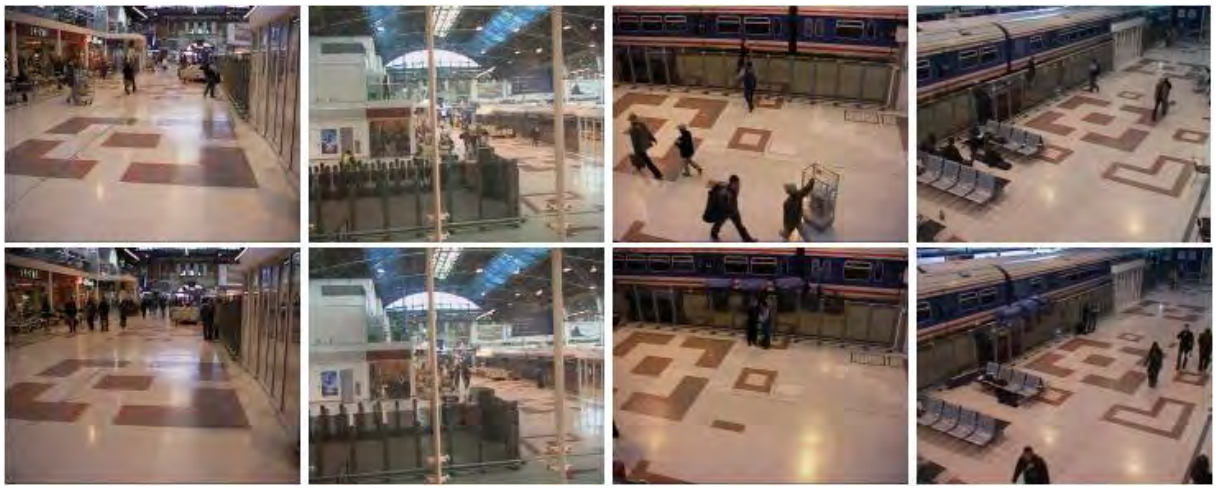


Figure B.4: Sample Frames of PETS 2006

B.4.3 PETS 2006

- Description: Multicamera person and baggage detection in a train station. Scenarios of increasing complexity, captured using multiple sensors.
- Number of sequences: 7 sets with 4 cameras each.
- Formats (for each set): MJPEG movies and JPEG frames.
- Tracking ground-truth available: no.

B.4.4 PETS 2007

- Description: multicamera setup containing the following scenarios: loitering; attended luggage removal (theft) and unattended luggage with increasing scene complexity.
- Number of sequences: 1 training set + 9 testing sets.
- Formats (for each set): JPEG frames.
- Tracking ground-truth available: no.

B.4.5 PETS 2010

- Description: multicamera setup containing different crowd activities (these datasets are the same as used for PETS2009).
- Number of sequences: 1 training set + 3 testing sets.

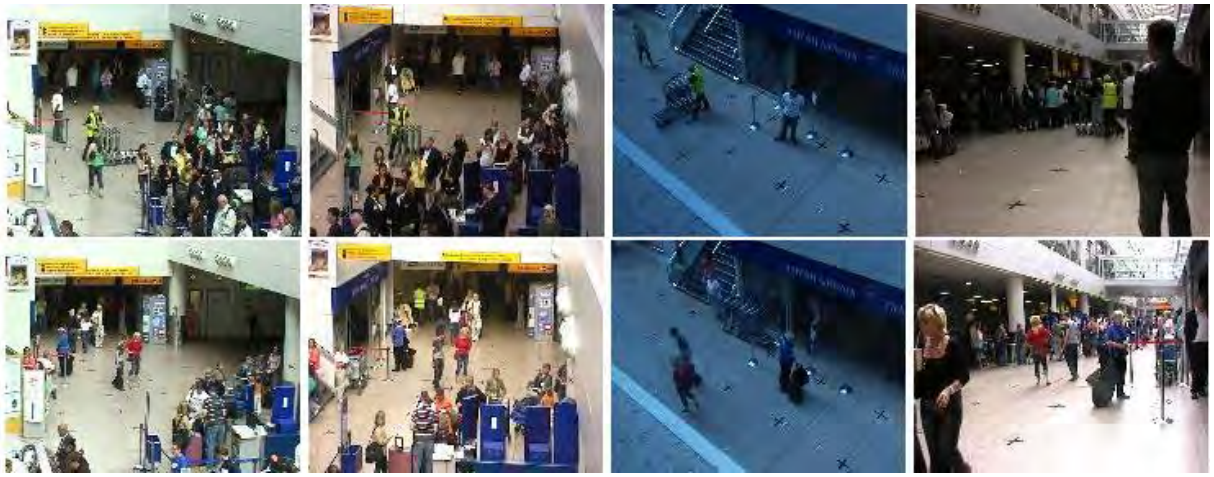


Figure B.5: Sample Frames of PETS 2007

Figure B.6: Sample Frames of PETS 2010

- Formats (for each set): JPEG frames.
- Tracking ground-truth available: no.

B.5 CAVIAR

The main objective of CAVIAR [68] is to address the scientific question: Can rich local image descriptions from foveal and other image sensors, selected by a hierarchical visual attention process and guided and processed using task, scene, function and object contextual knowledge improve image-based recognition processes [58]. Several methods were researched in order to address this question, including different areas, and the results were integrated in a closed-loop object and situation recognition system.

Figure B.7: Sample Frames of CAVIAR

- Description: this dataset includes sequences of people walking alone, meeting with others, window shopping, entering and exiting shops, fighting and passing out and leaving a package in a public place. All video clips were filmed with a wide angle camera lens, and some scenarios were recorded with two different points of view (synchronized frame by frame).
- Number of sequences: INRIA (1st set): 6 sequences, Shopping Center in Portugal (2nd set): 11 sequences, 6 different scenarios.
- Formats (for both sets): MJPEG movies, JPEG frames, XML ground-truth.
- Tracking ground-truth available: yes.

B.6 VISOR

The VIdeo Surveillance Online Repository is an extensive database containing a large set of multimedia data and the corresponding annotations. The repository has been conceived as a support tool for different research projects [55].

Some videos are available publicly, however, most of them are restricted and can only be viewed after a registration. The videos in the database cover a wide range of scenarios and situations, including (but not limited to) videos for human action recognition, outdoor videos for

Figure B.8: Sample Frames of Visor

face detection, indoor videos for people tracking with occlusions, videos for human recognition, videos for vehicles detection and traffic surveillance...

B.6.1 Video for indoor people tracking with occlusions

- Description: this dataset includes several videos with a wide range of occlusions caused by objects or people in the scene. All of them include base annotations and some also include automatic annotations.
- Number of sequences: 6 sequences.
- Format: MJPEG movies.
- Tracking ground-truth available: no.

B.7 iLids

The Imagery Library for Intelligent Detection Systems (i-Lids) bag and vehicle detection challenge was included in the 2007 AVSS Conference [69].

- Description: this dataset includes several sequences for two separate tasks: first, an abandoned baggage scenario and second, a parked vehicle scenario.
- Number of sequences: 7 sequences (3 for Task 1, 4 for Task 2)..
- Format: JPEG images, 8-bit color MOV, XML for ground-truth.
- Tracking ground-truth available: no.

Figure B.9: Sample Frames of i-Lids

B.8 Clemson dataset

Included in an elliptical head tracking project by Stan Birchfield there is a series of videos very interesting for head tracking. The sequences include issues such as occlusion, rotation, translation, clutter in the scene, change in the target's size, etc. The tracker as well as the sequences can be found at the web [70].

- Description: this dataset includes several sequences for head tracking with different targets. The videos include some of the most important issues for tracking algorithms.
- Number of sequences: 16 short sequences (1350 frames in total).
- Format: BMP images.
- ground-truth available: yes.

B.9 MIT Traffic Dataset

MIT traffic dataset [51] is for research on activity analysis and crowded scenes. It includes a traffic video sequence of 90 minutes long recorded by a stationary camera. The size of the scene is 720 by 480. More information regarding this work can be found in [71].

- Description: this dataset includes several clips regarding traffic. It contains a representation of most of the issues previously described, making this a very interesting dataset.
- Number of sequences: 1 sequence, 165880 frames divided in 20 clips.

Figure B.10: Sample Frames of Clemson

Figure B.11: Sample Frames of MIT Traffic Dataset

- Format: AVI clips.
- Tracking ground-truth available: yes, only for pedestrians.

Appendix C

Dataset Sequences Description

This appendix provides a detailed description of the sequences selected for composing the levels 3 and 4 of the proposed dataset.

C.1 Level 3

C.1.1 Cars

All sequences from this sublevel were obtained from MIT Traffic dataset [51] and Karlsruhe Cars dataset [52].

C.1.1.1 Basic

1) l3_cars_basic: a basic sequence from the MIT Traffic dataset where a red car moves from left to right without any issues.

C.1.1.2 Illumination Abrupt

1) dtneue_nebel_taxi (S1): a clip from the sequence *dtneue_nebel* from the Cars Karlsruhe dataset where a taxi moves throughout the scene from normal to under illuminated areas. The complexity of this sequence is low.

2) mv2_001_darkcar (S2): a clip from the sequence mv2_001 from the MIT Traffic dataset where a dark car moves throughout the scene, experiencing changes in the illumination as it goes through less illuminates areas. The complexity of this sequence is medium.

3) mv2_003_whitetruck (S3): a clip from the sequence mv2_003 from the MIT Traffic dataset where a dark car moves throughout the scene, experiencing changes in the illumination as it goes through less illuminates areas.The complexity of this sequence is high.

C.1.1.3 Noise

1) dtneu_schnee_redcar (S1): a clip from the sequence dtneue_schnee from the Cars Karlsruhe dataset where a red car moves through the scene while heavy snow falls.

2) Series of 3 sequences where the basic sequence has been altered to include a Gaussian noise (mean 0 and changing variance) and constant throught all the sequence (S2, S3, S4).

C.1.1.4 Occlusion

1) mv2_002_redcar (S1): a clip from the sequence mv2_002 from the MIT Traffic dataset where a red car moves throughout the scene, experiencing slight occlusions due to the presence of trees. The complexity of this sequence is low.

2) mv2_005_silvercar (S2): a clip from the sequence mv2_005 from the MIT Traffic dataset where a silver car moves throughout the scene, experiencing slight occlusions due to the presence or trees in the sight line. Since the car color is closer to the one of the road, this scenario has a medium complexity.

3) mv2_003_blackcar (S3): a clip from the sequence mv2_003 from the MIT Traffic dataset where a black car moves throughout the scene at a higher speed, experiencing slight occlusions due to the presence of. Since the car's speed is high, this scenario has a high complexity.

C.1.1.5 Scale Changes

1) mv2_001_whitecar (S1): a clip from the sequence mv2_001 from the MIT Traffic dataset where a small white car moves further away from the camera. The complexity of this sequence is medium.

2) mv2_003_redcar (S2): a clip from the sequence mv2_003 from the MIT Traffic dataset where a big red truck moves further away from the camera. The complexity of this sequence is medium.

3) mv2_003_whitevan (S3): a clip from the sequence mv2_003 from the MIT Traffic dataset where a big white truck moves further away from the camera. The complexity of this sequence is low.

C.1.1.6 Similar Objects

1) mv2_002_darkcar (S1): a clip from the sequence mv2_002 from the MIT Traffic dataset where a black car similar to the target moves near it for a few frames.

2) mv2_004_darkcar (S2): a clip from the sequence mv2_004 from the MIT Traffic dataset where a black car moves close to two other similar cars.

3) mv2_006_redtruck (S3): a clip from the sequence mv2_006 from the MIT Traffic dataset where a red truck moves close to a red car for a few frames.

C.1.2 Faces

All sequences from this sublevel were obtained from TRECVID 2009 [53], CLEMSON dataset [54] and VISOR [55]. An example of the faces sequences can be found in Figure 4.7.

C.1.2.1 Basic Sequence

As explained in the previous level, the basic sequence is not used in the evaluation, but some alterations are added in order to create noise and illumination changes sequences. This basic sequence is a clip of MCTTR0205a.mov.deint (TRECVID2009) where an old man walks slowly from one side of the scene to the other.

C.1.2.2 Complex Movement

1) visor1_man_head (S1): a clip from visor_1206627910990_video_1 of VISOR dataset where the target changes its position abruptly.

C.1.2.3 Illumination Gradual

1) Series of 3 sequences where the basic sequence has been altered to include an increase in the pixel intensity, that is, to brighten its illumination (S1, S2, S3).

C.1.2.4 Illumination Abrupt

1) Series of 3 sequences where the basic sequence has been altered to include an abrupt change in the pixel intensity covering half of the image (S4, S5, S6).

C.1.2.5 Noise

1) Series of 3 sequences where the basic sequence has been altered to include a Gaussian noise (mean 0 and changing variance) and constant throught all the sequence (S2, S3, S4).

C.1.2.6 Occlusion

1) visor2_man_head (S4): a clip from visor_1206627914419_video_2 from VISOR dataset where the target is occluded totally by another person walking in front of him.

2) visor5_man_head (S5): a clip from visor_1206627921666_video_5 from VISOR dataset where the target is occluded totally by a blue board with different entry and exit points.

3) visor6_man_head (S6): a clip from visor_1206627923895_video_6 from VISOR dataset where the target is occluded totally and several times by a blue board with different entry and exit points.

C.1.3 People

All sequences from this sublevel were obtained from the following datasets: PETS 2009 [56], TRECVID 2009 [53], i-Lids [57], CAVIAR [58] and PETS 2000 [59].

C.1.3.1 Basic Sequence

As explained in the previous level, the basic sequence is not used in the evaluation, but some alterations are added in order to create noise and illumination changes sequences. This basic sequence is a clip of a video from the 2009 PETS dataset (PETS2009_S2_L1_view001).

C.1.3.2 Complex Movement

1) MCTTR0205a_blueman (S2): a clip from MCTTR0205a.mov.deint (TRECVID2009) where a man in a blue jumper walks fast from one side to the other of the image.

2) MCTTR0205a_darkman (S3): a clip from MCTTR0205a.mov.deint (TRECVID2009) where a man in a dark suit walks fast from one side of the image to the other. There is a slight change of scale but it was not taken into account.

3) MCTTR0205a_suitcaseman (S4): a clip from MCTTR0205a.mov.deint (TRECVID2009) where a man with a suitcase walks fast from one side of the image to the other. There is also a slight change of scale but it was not taken into account either.

C.1.3.3 Illumination Gradual

1) Series of 3 sequences where the basic sequence has been altered to include an increase in the pixel intensity, that is, to brighten its illumination (S1, S2, S3).

C.1.3.4 Illumination Abrupt

1) Series of 3 sequences where the basic sequence has been altered to include an abrupt change in the pixel intensity covering half of the image (S7, S8, S9).

C.1.3.5 Noise

1) Series of 3 sequences where the basic sequence has been altered to include a Gaussian noise (mean 0 and changing variance) and constant throught all the sequence (S5, S6, S7).

C.1.3.6 Occlusion

1) AB_Easy_man (S7): a clip from AB_Easy (iLids) where a man walking through the scene is occluded by a column.

C.1.3.7 Scale Changes

1) AB_Easy_manbag (S4): a clip from AB_Easy (iLids) where a man with a backpack walks further from the camera, decreasing his size.

2) AB_Hard_whiteman (S5): a clip from AB_Hard (iLids) where a man dressed in a white jumper moves further away from the camera.

3) CAVIAR_trespasillo_redman (S6): a clip from CAVIAR_ThreePastShop2cor where a man in a red coat walks in a long corridor, decreasing his size.

4) PETS2009_S2_L1_girl (S7): a clip from PETS2000_S2_L1 where a girl walks around the scene and moves further away from the camera.

C.1.3.8 Similar Objects

1) AB_Hard_girl (S4): a clip from AB_Hard (iLids) where a girl in a black coat walks around the scene while other people dressed with the same colors move near her.

2) CAVIAR_trespasillo_man (S5): a clip from CAVIAR_ThreePastShop2cor where a man in a black coat moves close to the target, also wearing a black coat.

3) PETS2009_S2_L2_jeans (S6): a clip from PETS2009_S2_L2 where a man in jeans and a black sweater walks around the scene with other people dressed in similar colors.

C.2 Level 4

C.2.1 Cars

- mv2_020_red (S1): includes scale changes (the car moves further away) as well as illumination changes.
- mv2_020_silver (S2): includes scale changes (the car moves further away) and several illumination changes.

- mv2_020_whitevan (S3): includes similar objects, appearance changes, scale changes and illumination changes.

C.2.2 Faces

- HEADTRACK_seq_bb (S1): includes rotation (which can be viewed as an occlusion) and distraction (similar objects in background)
- HEADTRACK_seq_jd (S2): includes a severe occlusion by a similar object.
- HEADTRACK_seq_mb (S3): includes overlapping of two faces as well as similar objects in the background (flesh-colored board and boxes)
- HEADTRACK_seq_ms (S4): includes a hand movement intended to distract the tracker.
- HEADTRACK_seq_sb (S5): includes target rotation and tilting, occlusion and similar objects.
- HEADTRACK_seq_villains2 (S6): includes three similar objects intended to distract the tracker from the real target.

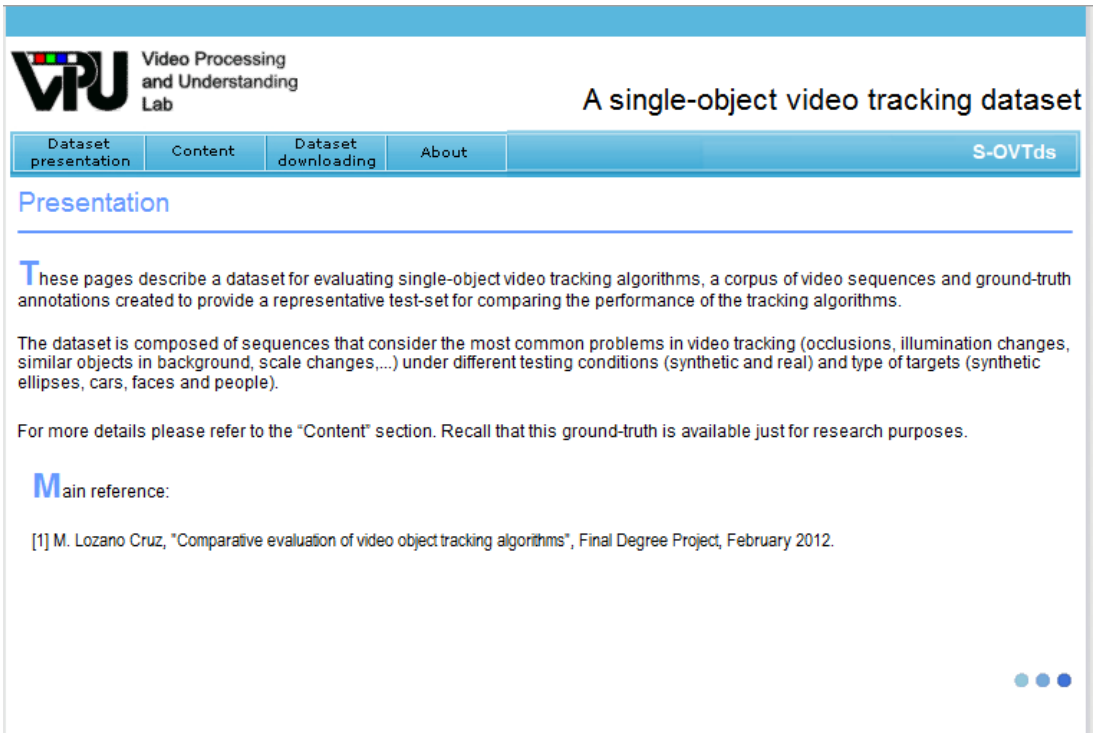
C.2.3 People

- PETS2009_S2_L1_view001_1 (S1): includes similar objects, partial and total occlusions and slight scale changes.
- PETS2009_S2_L1_view001_2 (S2): includes similar objects, occlusions and scale changes.
- PETS2009_S2_L2_view001_1 (S3): includes illumination changes, occlusions and scale changes.
- PETS2009_S2_L2_view001_2 (S4): includes illumination changes and scale changes.
- PETS2009_S2_L3_view001_1 (S5): includes illumination changes, occlusions and scale changes
- PETS2009_S2_L3_view001_2 (S6): includes illumination changes and scale changes.

Appendix D

Web Page

The proposed dataset is available on Internet at the following website `http://www-vpu.ii.uam.es/SOVTds/index.html`. In this section, we include some snapshot of the website:



Video Processing and Understanding Lab

A single-object video tracking dataset

Dataset presentation | Content | Dataset downloading | About | S-OVTds

Content

Level L1

>Tracking dataset

Level L1

Level L2

Level L3

Level L4

Description

This level shows different synthetic scenes generated with Matlab with an elliptical target. There are five sequences for each problem, covering a wide range of complexities.

Number of sequences

35 sequences

Isolated tracking-related problems

5 sequences each, with varying degree of complexity, from low to high: Complex Movement, Global Illumination Changes, Local Illumination Changes, Noise, Occlusion, Scale Changes, Similar Objects.

Examples

Low Complexity

Sequence name

l1_noise_1.avi

Tracking-related problem

Noise

Length

100 frames.

Example frames: 1, 50, 100.

Sequence preview

Appendix E

Initialization Errors vs. Issues

This appendix described the effect of incorrect initialization in the sequences that compose the proposed dataset which describes the common issues in video tracking. The following data depicts the tracking results regarding the analyzed sequences with (a) correct initialization (indicated by ground-truth data), (b) modifications of the initialization dimensions, (c) modifications of initialization center (i.e., traslation), and (d) modifications of the center and dimensions of the initialization. The complexity indicated in each figure represents that at least 80% (Low), 50% (Medium) or 30% (High) of the modified bounding box overlaps with the ground-truth annotation.

E.1 Complex movement

The obtained results are depicted in Figure E.1. As it can be observed in Figure E.1 (b), the most affected algorithms are MSA and TM, the later being more dependent of the complexity. For the other algorithms,slight size changes do not seem to highly affect the performance. Figure E.1 (c) shows that very good results are obtained for the first two complexity groups (80% and 50% Overlap) , although a slight decrease in the accuracy is observed for the high complexity scenario (30% Overlap). Figure E.1 (d) shows that, for high complexity, all algorithms experience a significant decrease in their accuracy. Globally, all algorithms presented a performance decrease against initialization modifications in presence of complex target motion. Also note that some algorithms (such as CBWH) performed better with a slight increase of the size of the target demonstrating the existence of errors in the ground-truth annotations.

E.2 Illumination gradual

Results for sequences with gradual illumination changes are shown in Figure E.2. The results for the three complexity levels are very similar results for the three cases (b), (c) and (d), and all of

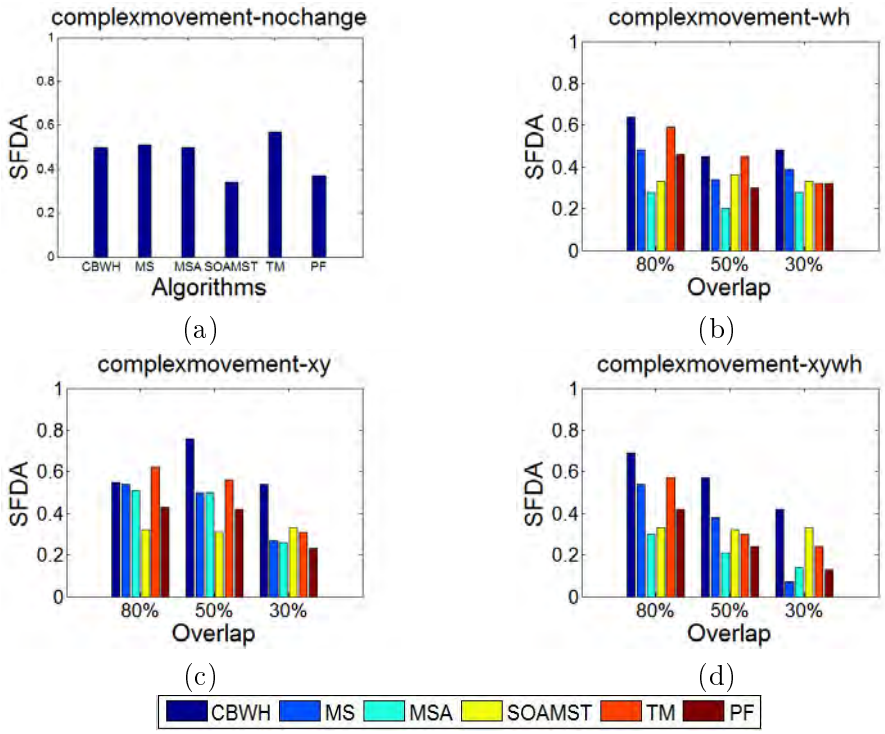


Figure E.1: Complex Movement Initialization Comparison

them are very similar to the accuracy values obtained when the box experiences no modification (a) for this issue. In all cases, a slight accuracy decrease is experienced by all algorithms as the complexity increases respect to the correct initialization case (Figure E.2(a)). Moreover, SOAMST exhibited high robustness to changes in the target initialization whilst the rest of the algorithms decreased their performance as the complexity increases.

E.3 Illumination abrupt

The results for sequences with abrupt illumination changes are depicted in Figure E.3. Figure E.3(b) shows very similar values for all algorithms except MSA, which experiences an abrupt decrease as the complexity increases, which also occurs in previous cases. Figure E.3(c) demonstrated that all algorithms have similar performance to the correct initialization case (Figure E.3(a)). Figure E.3(d) shows a gradual accuracy decreases the complexity increases, being the MSA the most noticeable algorithm, with the exception of SOAMST that remained invariant to the initialization modifications.

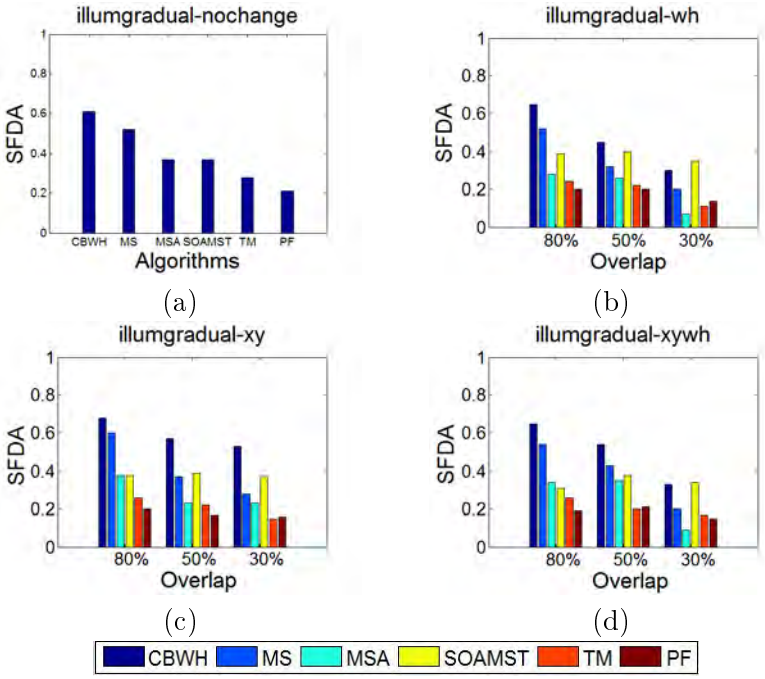


Figure E.2: Illumination Gradual Initialization Comparison

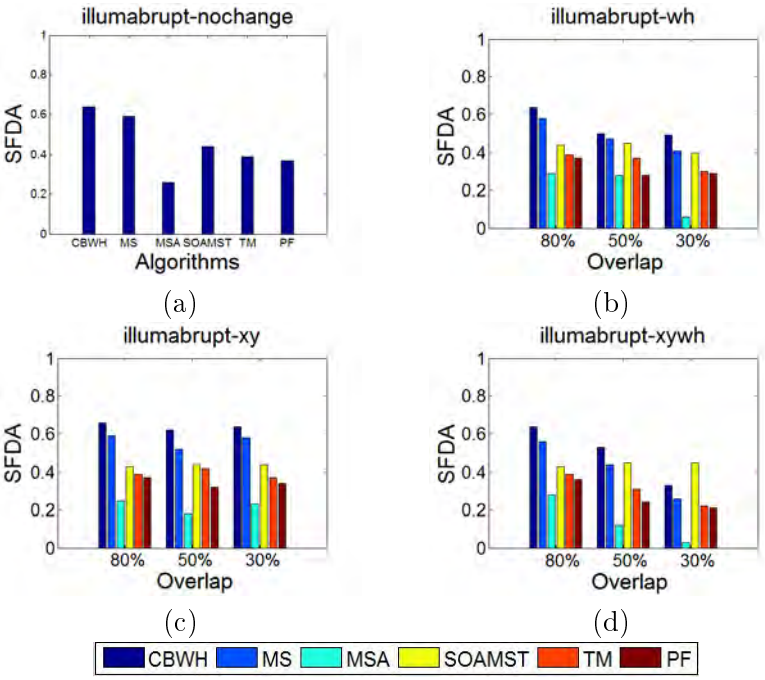


Figure E.3: Illumination Abrupt Initialization Comparison

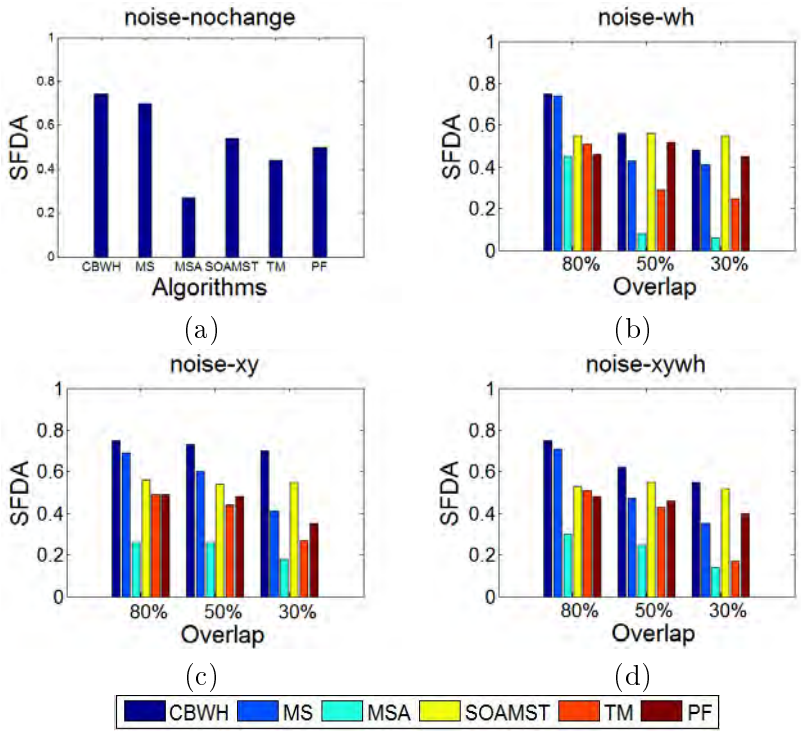


Figure E.4: Noise Initialization Comparison

E.4 Noise

Results for sequences with different noise levels are depicted in Figure E.4 where most algorithms achieve very high accuracy. In Figure E.4 (b), a slight and gradual decrease of accuracy (especially CBWH and MS) is observed. MSA decreased very abruptly, with extremely poor results as the complexity increases. On the other hand, SOAMST and PF remained invariant to this modification. In Figure E.4 (c), a slight decrease for all algorithms is observed except for CBWH and SOAMST. Finally, Figure E.4 (d) shows that the same decrease pattern is observed, with all algorithms except SOAMST providing lower accuracy values as the complexity increases.

E.5 Occlusion

Results for sequences including some sort of target occlusion are depicted in Figure E.5. The similarity between all cases (b), (c) and (d) is obvious, and the most noticeable difference is the decrease in the performance of MSA both when the size of the box changes and when the size and position change. In this case, it is possible to say that all algorithms (with the exception of MSA) are robust to initialization errors when the sequences have occluded targets.

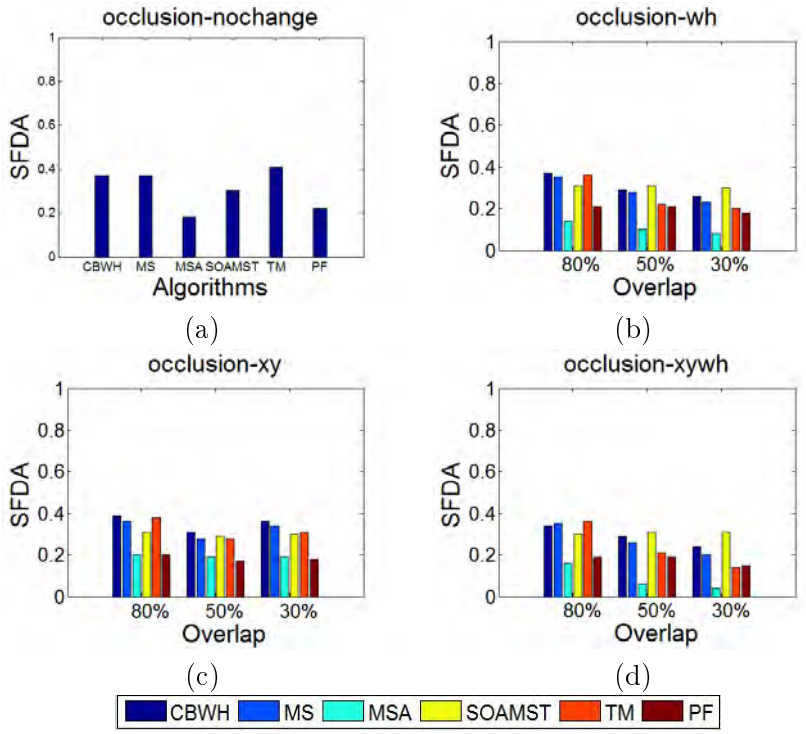


Figure E.5: Occlusion Initialization Comparison

E.6 Scale changes

The results for sequences where the target experiences scale changes are shown in Figure E.6. In Figure E.6 (b), algorithms had a performance decrease as the initialization size increase in complexity with the exception of SOAMST and PF, which remained constant in all three complexity levels. In Figure E.6 (c), both CBWH and SOAMST were independently to the initialization errors, whereas the rest of the algorithms presented a gradual and slight decrease. In Figure E.6 (d), all algorithms (with the exception of SOAMST) experienced worst results as the complexity of the initialization is increased.

E.7 Similar objects

In Figure E.7, results for sequences where similar objects to the target appeared in the sequence are depicted. In the three cases, all algorithms experienced a gradual and slight decrease as the initialization error increased, with the exception of SOAMST, which provided constant results for all complexity levels.

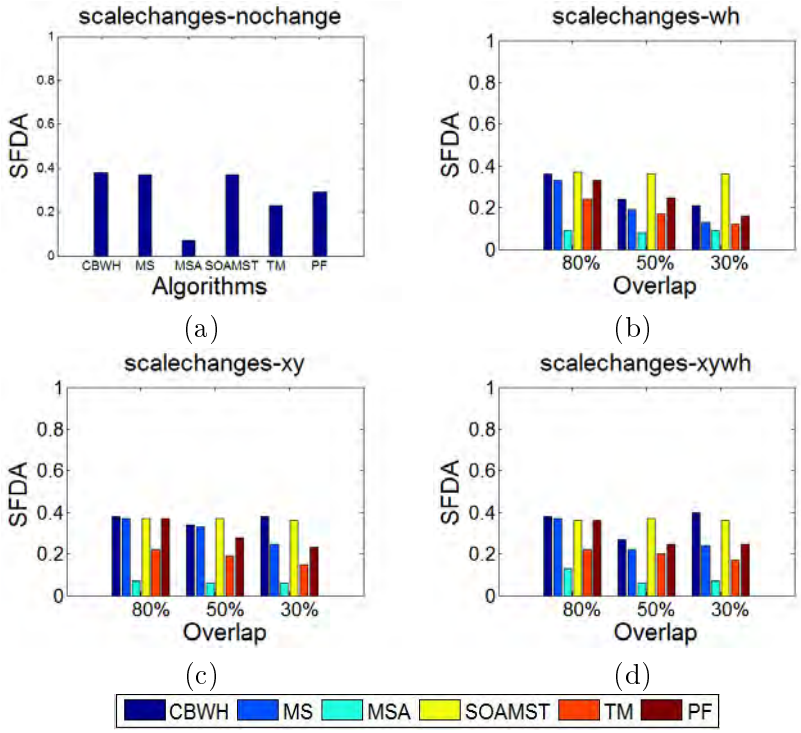


Figure E.6: Scale Changes Initialization Comparison

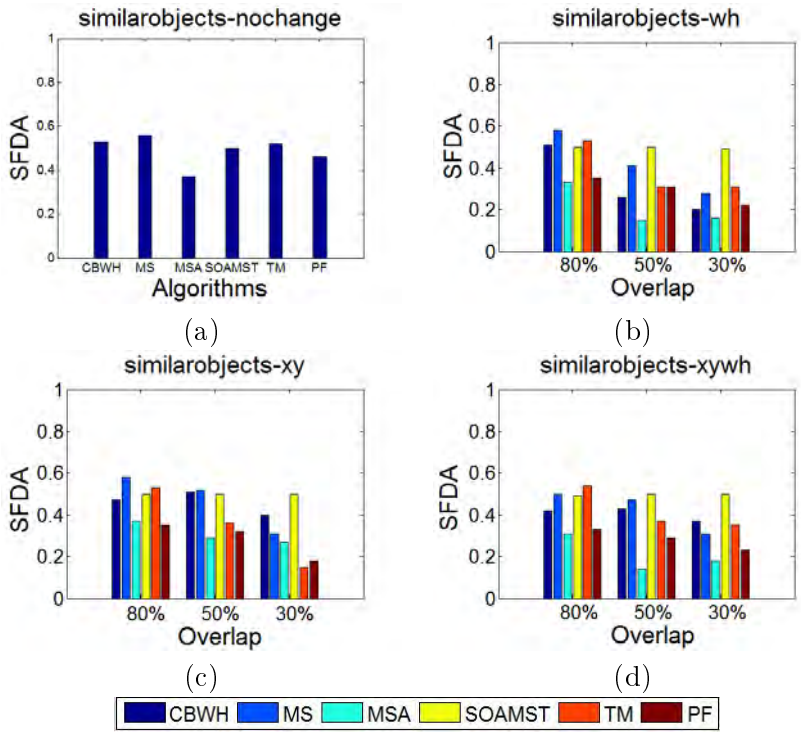


Figure E.7: Similar Objects Initialization Comparison

Appendix F

Introducción

Este capítulo introduce el trabajo presentado en este documento. En las próximas secciones, se describe la motivación del trabajo (sección F.1), los objetivos principales (sección F.2) y la estructura del documento (sección F.3).

F.1 Motivación

La visión artificial es un campo cuyo objetivo es el de automatizar el procesamiento de imágenes (tomadas, por ejemplo, con una cámara o un conjunto de cámaras) para entender su contenido. La visión artificial trata de imitar el sistema de visión humano en el que el cerebro procesa imágenes capturadas por los ojos [1]. Los datos pueden tener diversos formatos, como por ejemplo, secuencias de video, diferentes vistas de múltiples cámaras o datos multidimensionales proporcionados por escáneres medicos. Esta información es usada a continuación para resolver tareas o entender que sucede en la escena representada. Este campo tiene diversas aplicaciones en las areas de visión industrial (por ejemplo, inspección de partes mecánicas), detección de eventos (por ejemplo, detección de equipajes abandonados) y aplicaciones forenses y biométricas (por ejemplo, reconocimiento facial automático).

El seguimiento de objetos en video[1] es un paso importante en muchas de las aplicaciones relacionadas con la visión artificial. Consiste en la localización del objeto (u objetos) de interés[2] según se mueve en el tiempo a través de una escena por medio de un dispositivo de visión como puede ser una cámara [2]. Encontrar el objeto de interés en frames consecutivos puede ser difícil si se mueve rápido con respecto al frame rate de la secuencia de video. Una solución típica para este problema es el uso de un modelo de movimiento para describir la dinámica del objeto. Un

[1] En este documento se utiliza el término *tracking* o *video tracking* para referirse al seguimiento de objetos en una secuencia de video.

[2] En este documento se utiliza el término *target* para representar los objetos de interés que serán seguidos en una secuencia de video.

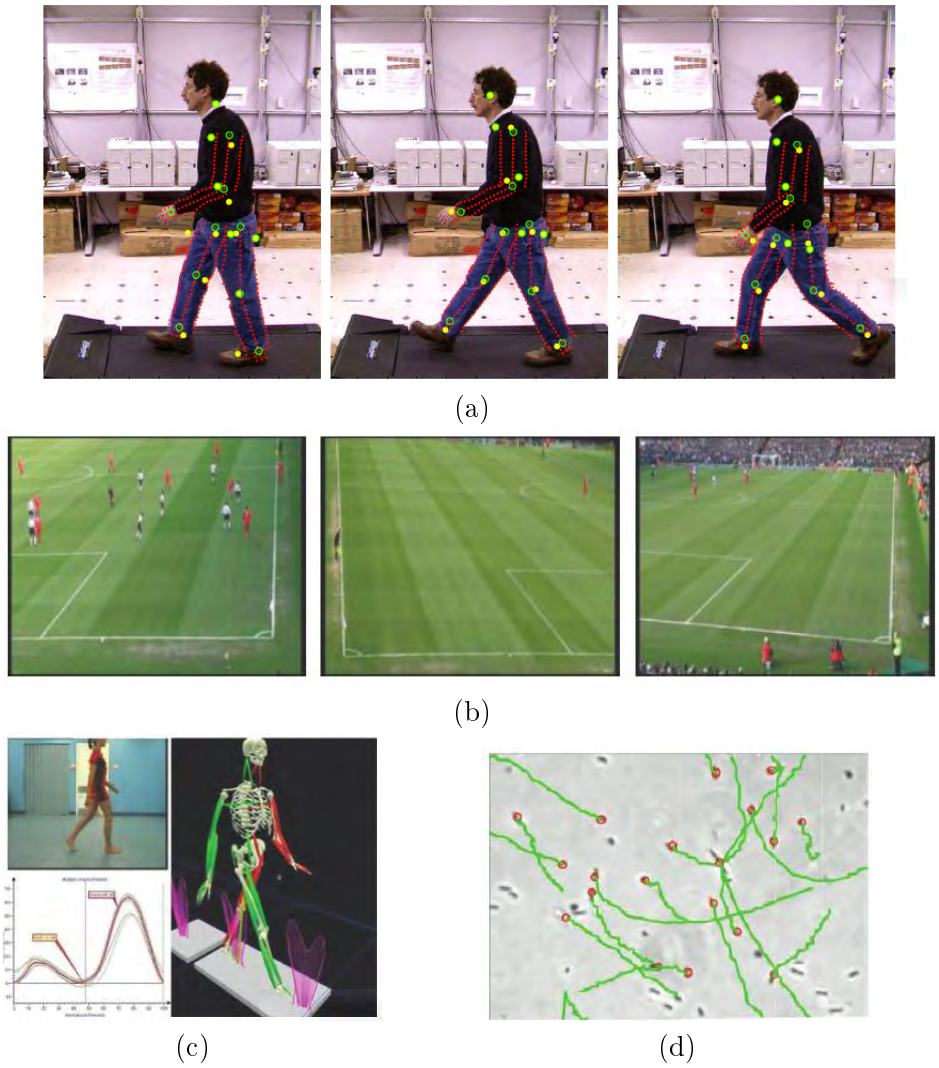


Figure F.1: Examples of video tracking: (a) motion capture analysis [3], (b) sports tracking with multiple cameras (from PETS2003 dataset), (c) gait analysis (Oxfords Metrics Group) and (d) position tracking of *Escherichia coli* bacteria (from [4])

amplio rango de aplicación surgen del seguimiento de objetos en videos, por ejemplo, interacción humano-máquina, seguridad y vigilancia, comunicación por video, realidad aumentada, control de tráfico, radiodiagnostico y editado de videos: algunos ejemplos estan representados en la Figura F.1. Dado que la cantidad de datos a tratar es muy grande, el seguimiento de objetos es una tarea de gran complejidad y con una gran consumo de tiempo. Esta complejidad puede además verse aumentada debido al hecho de que suele ser necesario emplear técnicas de reconocimiento de objetos.

El diseo de un algoritmo de seguimiento de objetos es una tarea complicada. Se ha llegado al acuerdo de que los tres pasos para el diseño de un algoritmo son [2]:

1) Extracción de información relevante: identificación y extracción de las características mas importantes del *target* que serán usadas en el *tracking.* Cuanto mejor sea esta selección, más robusto sera el algoritmo.

2) Representación del *target*: este paso define el modelo con la información importante extraída por el algoritmo de seguimiento. El método ideal de representación permite identificar el objeto sin duda siendo lo suficientemente flexible como para soportar cambios de escala, orientación, iluminación...

3) Propagación del modelo del *target* en el tiempo, utilizando recursivamente información de pasos previos.

Una gran variedad de técnicas han sido desarrolladas siguiendo los pasos mencionados arriba. En este contexto, la selección del algoritmo mas adecuado para cada aplicación es una tarea de gran complicación y que normalmente lleva a cabo el diseñador de la aplicación basado en su experiencia. Por otra parte, la alta variabilidad y complejidad de los datos a analizar tiene que tenerse en cuenta en esta selección. Hay varios problemas que afectan al rendimiento del algoritmo, como por ejemplo ruido, objetos similares en el fondo, oclusiones, cambios de apariencia o iluminación... Por lo tanto, no hay un único algoritmo que se comporte perfectamente en todas y cada una de las situaciones.

Para identificar claramente qué algoritmos se comportan mejor en determinadas situaciones o aplicación, se ha propuesto en la literatura una evaluación del rendimiento como un metodo de determinar tanto los puntos fuertes como los débiles de cada uno de los algoritmos. Dicha evaluación consiste en el análisis de los resultados obtenidos. Para llevar a cabo dicho analisis, hay que especificar dos aspectos fundamentales: el dataset (una serie de secuencias que cubren las situaciones y problemas a los que el algoritmo se va a enfrentar, y que es lo suficientemente extensiva para ser representativa del mundo real) y las métricas que se usarán para representar la precisión de los algoritmos de tracking (que permiten cuantificar cómo de bien se comportan los algoritmos). Estos dos aspectos tambien son conocidos como el protocolo de evaluación de seguimiento de objetos [2]. Además, los protocoles de evaluación mas comunes utilizan métricas basadas en información de ground-truth, que representa el resultado de tracking ideal y cuya anotación se hace manualmente. La generación del ground-truth es un paso muy lento y por lo tanto, suele limitar la cantidad de secuencias y su extensión en los datasets. Es mas, la existencia de diversas métricas incrementa la dificultad al diseñar un protocolo de evaluación exacto. Otro punto a tener en cuenta es la cantidad cada vez mayor de información disponible, lo que genera una nueva necesidad de automatizar todo el proceso de evaluacion de *tracking.*

El número de estudios comparativos llevados a cabo con diferentes algoritmos es muy limitados dado el bajo número de información empleada así como la similitud de las métricas analizadas. Por ejemplo, en [6] se proponían diversas metricas que generaban información redundante. Por

lo tanto, no es fácil extrapolar las conclusiones extraídas del análisis a nuevas secuencias. sonIt represents the ideal tracking result and it is manually annotated. The generation of the ground truth is usually a time consuming step and, therefore, limits the amount of data in the dataset. Furthermore, the existence of several metrics increases the complexity of designing an accurate evaluation protocol. Another point to be taken into account is the increasing quantity of video data available, which generates a new need to automate the whole tracking evaluation process. Otro problema a tener en cuenta es el hecho de que no hay estudios comparativos de algoritmos de diversa naturaleza cuando los problemas estan presentes en las secuencias. Este hecho también limita las conclusiones extraídas de las evaluaciones de rendimiento.

Como se ha explicado anteriormente, los datasets disponibles actualmente no son suficientes para cubrir las necesidades actuales de los aplicaciones de seguimiento de objetos [**?**]. Actualmente, no existe un protocolo de evaluación estándar que permita una comparación rápida e intuitiva de algoritmos, mostrando tanto las ventajas como las desventajas de cada uno de ellos al enfrentarse a diversas situaciones. Dado que cada protocolo de evaluación propone diferentes datasets y metricas, no es posible tener una visión general de los resultados que permita la comparación de diferentes algoritmos en diversos escenarios.

F.2 Objetivos

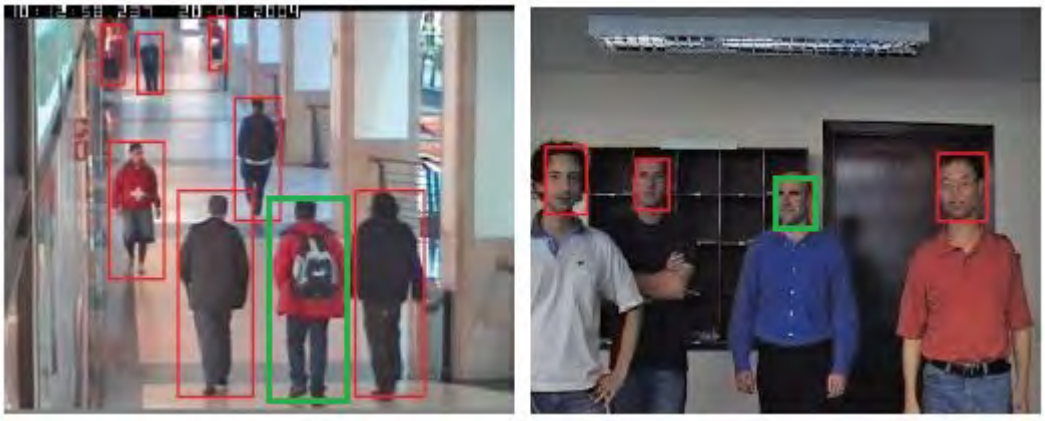
El objetivo principal del trabajo presentado en este documento es el desarrollo de un protocolo de evaluación para estimar el rendimiento de algoritmos de seguimiento de objetos. Este protocolo debe tener en cuenta las complicaciones o problemas más destacas a las que se ven enfrentadas los algoritmos. Para ellos, los siguientes puntos han sido realizados:

- Estudio en profundidad del estado del arte. Incluye un análisis del trabajo relacionado con seguimiento de objetos teniendo en cuenta los diferentes estados del análisis, los métodos actuales y los problemas más comunes, asi como las métricas de evaluación y los datasets disponibles.

- Selección e implementación de las técnicas de *tracking* más representativas. Se seleccionan las técnicas más representativas de algoritmos deterministas y probabilísticos propuestas en la literatura.

- Creación de un dataset apropiado para la evaluación del *tracking*. Este paso consiste en el diseño de secuencias tanto sintéticas como reales que representen los problemas más importantes. También se usarán secuencias reales disponibles en otros datasets.

- Diseño e implementación de un protocolo de evaluación del rendimiento del *tracking*. El objetivo es proponer una metodología para evaluar algoritmos que cubra los diferentes tipos de secuencias así como los problemas más habituales.



(a)

(b)

Figure F.2: (a) Video tracking examples and (b) issues in tracking.

- Aplicación del protocolo de evaluación propuesto. El rendimiento de los algoritmos seleccionados es puesto a prueba usando el protocolo previamente definido e implementado.

F.3 Estructura del Documento

El documento está estructurado de la siguiente manera: The structure of the document is as follows:

- Capítulo 1. Este capítulo presenta la motivación y los objetivos del trabajo de este documento.
- Capítulo2. Este capítulo analiza la literatura relacionada con el trabajo expuesto en este documento.
- Capítulo3. Este capítulo describe los diferentes enfoques (algoritmos) que serán evaluados para seguir un objeto.

- Capítulo4. Este capítulo describe el protocolo de evaluación presentados. Incluye tanto los aspectos que serán considerados como el dataset empleado.
- Capítulo5. Este capítulo presenta un análisis en profundidad de la aplicación del protocolo de evaluación en los algoritmos previamente seleccionados.
- Capítulo6. Este capítulo resume los resultados obtenidos en este trabajo y presenta algunas sugerencias de trabajo futuro.

Appendix G

Conclusiones y Trabajo Futuro

G.1 Resumen

En este documento se ha presentado un nuevo protocolo de evaluación para el seguimiento de objetos en video, permitiendo la evaluación de diferentes casos y proporcionando un marco con el que comparar el rendimiento de los algoritmos de seguimiento.

El primer paso fue realizar un extensivo estudio acerca del estado del arte, necesario para entender todo el proceso de *video tracking*[1]. También se realizó un estudio de las métricas existentes y los datasets disponibles para la evaluación de *video tracking*. Este estudio se encuentra disponible en el capítulo 2.

A continuación se realizó una completa búsqueda y analisis de diversos algoritmos. Una vez que se hubo entendido el funcionamiento de los algoritmos (basados en color), se hizo una selección de los mas relevantes. La idea principal era utilizar algoritmos que abarcaran diferentes enfoques: incluyendo información de fondo, capaces de hacer frente a cambios de escala y orientación y con enfoques determinísticos y probabilísticos. Todos ellos tienen en común el uso de un rasgo: el color. Toda la información acerca de los algoritmos seleccionados se encuentra en el capítulo 3.

Una vez que los algoritmos habían sido seleccionados e implementados fue necesario desarrollar un protocolo para su evaluación teniendo en cuenta diversos aspectos: estudio de parámetros óptimos, robustez a errores de inicializacion, precisión, estabilidad y eficiencia. Para la evaluación de precisión (*accuracy*), el número de métricas existentes es abrumador, sin embargo, tras un cuidado estudio, se hizo evidente que la mayoría de las métricas proporcionaban información redundante o no proporcionaban resultados fiables debido a detalles omitidos en sus definiciones. Se realizó entonces una selección meditada para utilizar las métricas que proporcionaran buenos resultados e incluirlas en el protocolo de evaluación para evaluar las secuencias. Una vez que

[1]Se usara el término *video tracking* en lugar de la traducción en español, seguimiento de objetos en video.

tanto los algoritmos como las métricas estuvieron listos, surgió la necesidad de crear un completo dataset. A la hora de diseñarlo, el objetivo principal era cubrir la mayor cantidad de situaciones y problemas posibles, y despues de un estudio intensivo de los datasets existentes, se seleccionaron, crearon y anotaron un total de 122 secuencias. Todo este trabajo se encuentra detallado en el capítulo 4.

En último lugar, el protocolo completo fue probado para comprobar su funcionamiento a la hora de evaluar los algoritmos seleccionados. El primer objetivo era comprobar cómo se comportaban los algoritmos cuando determinados problemas aparecían en las escenas así como el funcinamiento cuando ocurrían otros problemas (por ejemplo, mala inicialización). Este paso permitió determinar qué algoritmo (teniendo en cuenta que sólo se realizó una implementación de cada algoritmo) se comportaba mejor y proporcionaba mejores resultados no sólo en cada caso por separado sino con el dataset completo. El segundo objetivo era determinar otros detalles como el consumo de tiempo de cada algoritmo, cómo se veían afectados los algoritmos al modificar algunos de sus parámetros o cómo se veía afectado el funcionamiento con diferentes enfoques (probabilístico o determinístico). Toda la información acerca de los resultados experimentales obtenidos tras un extensivo análisis del protocolo se encuentran en el capítulo 5.

G.2 Conclusiones

Tras analizar los resultados obtenidos en el capítulo 5, se puede determinar que el mejor algoritmo es CBWH, ya que tuvo la mayor tasa de éxito en el dataset completo. Por otro lado, también tiene tiempos de ejecución menores que la mayoría de algoritmos, algo que es crucial cuando se analizan secuencias largas. MS obtuvo resultados similares a CBWH en cuanto a tasa de éxito con tiempos de ejecución algo mejores. El peor algoritmo es SOAMST sin lugar a dudas: baja tasa de éxito, comportamiento errático y alto tiempo de procesado. Sin embargo, este algoritmo proporcionó los resultados más estables en secuencias cuya anotación tenía algún tipo de error en la inicialización (en posición, tamaño o una combinación de ambos), debido al hecho de que el área de búsqueda dinámica permite al algoritmo ajustarse para encontrar el *target* con mayor precisión. La versión adaptada de MS, MSA, no proporcionó resultados favorables ya que la actualización del modelo suponía que en el momento en que se producía una ligera inclusión de características no deseables el algoritmo perdía fácilmente el *target* sin ser capaz de encontrarlo de nuevo. Por último, PF se comportó peor de lo esperado, que como se comentó anteriormente, puede ser debido al hecho de que las métricas empleadas no son las más apropiadas para este algoritmo, o a que no es posible configurar óptimamente los parámetros en un dataset con sequencias tan dispares y diferentes entre sí. Todas las conclusiones, resultados y comparaciones son válidas para este dataset específico.

En las próximas secciones se expondrán los resultados obtenidos con respecto a diferentes

características de los algoritmos seleccionados.

Modelo fijo En este caso tenemos dos algoritmos: MS y TM. Ambos son métodos simples, pero mientras que TM busca una imagen fija comparando la intensidad de los píxeles, MS utiliza histogramas (de color en este caso) para encontrar una mejor equivalente. Esta mejora se muestra claramente en los resultados: MS tiene un comportamiento mucho mejor en general. La desventaja de MS frente a TM es que consume el doble de tiempo. En algunos casos como movimiento complejo, cambios abruptos de iluminación u objetos similares en la escena, ambos algoritmos presentan un comportamiento similar. En el caso de oclusiones, TM es una mejor opción debido al hecho de que el área de búsqueda es la imagen completa y por lo tanto, una vez que el *target* es visible de nuevo (la oclusión termina), TM consigue encontrarlo mas rápido que otros algoritmos (como MS), pero MS tiene un comportamiento mejor en cambios graduales de iluminación, ruido, cambios de escala, y más importante, secuencias complejas con varios problemas debido a las limitaciones obvias de usar un modelo fijo (TM) frente a un histograma de color (MS).

Modelo adaptado En este caso, se realizó una modificación de MS para que el algoritmo tuviera en cuenta la información del frame anterior: en lugar de utilizar un modelo fijo a partir de la información del primer frame, el modelo se creaba en cada frame. Sin embargo, a pesar de que esto puede parecer una ventaja, el algoritmo propociona resultados mucho peores ya que se acumulan los errores a lo largo del proceso de seguimiento (*drift*). Por lo tanto, un modelo ciego (*blind adaptative scheme model*) no está recomendado.

Información de fondo El algoritmo CBWH, una mejora de MS, utiliza información del fondo (*background*), proporcionando los mejores resultados en general para todo el dataset. La única desventaja es, como ya se ha comentado, el mayor tiempo de ejecución debido a la necesidad de realizar más pasos en cada *frame*. Este problema puede solucionarse en parte disminuyendo el tamaño de ventana, aunque eso supondría una menor precisión (menor SFDA). En general, la inclusión de información del fondo es una adicción al algoritmo básico MS muy recomendable.

Información de tamaño En este caso tenemos dos algoritmos: SOAMST y PF. Ambos tratan los problemas de escalas desde su enfoque: punto de vista determinista (SOAMST) o probabilista (PF). En este caso, ambos algoritmos presentan resultados similares en cuanto a la tasa de éxito: en algunos casos SOAMST tiene un mejor comportamiento, mientras que en determinados problemas es PF el que propociona mejores resultados. En secuencias con una combinación de problemas (Level 4), PF es claramente el algoritmo que propociona mayor tasa de exito. Dado que estos algoritmos están preparados para tratar con los cambios de escala, es importante comentar que el comportamiento cuando se prueban secuencias con dicho problema

no es bueno: no sólo presentan resultados peores que otros algoritmos (como CBWH), sino que hay situaciones (como en objetos similares) en las que la precisión es mayor. Como apunte final, PF es el algoritmo más rápido mientras que SOAMST es el más lento: tarda unas 10 veces más que PF.

Los algoritmos que no incluyen tratamiento de cambios de escala se comportan bien en este dataset. Una posible razón es el hecho de que en los cambios de escala en los que el objeto se aleja de la cámara (reduce su tamaño), el algoritmo tiene mayores probabilidades de encontrarlo ya que su zona de búsqueda será muy grande, mejorando de esta manera la precisión general (a pesar de que el área será entonces mayor de lo necesario).

Enfoque determinista frente a probabilista Como se observa en la figura 5.23, hay una diferencia clara entre los mejores algoritmos (CBWH en la mayoría de los casos) frente PF. Por lo tanto, con este dataset y las implementaciones realizadas para cada algoritmo, así como las métricas empleadas, se puede decir que el enfoque determinstca proporciona resultados mejores que el probabilista. La mayor ventaja de PF es sin duda el menor tiempo de ejecución, mientras que la mejor característica de CBWH es la mayor precisión (valores altos de SFDA).

G.3 Trabajo futuro

Las conclusiones presentadas en el apartado anterior demuestran que algunos algoritmos operan de forma mucho mejor que otro y son, por tanto, una mejor opción a la hora de diseñar un sistema de *video tracking*. Sin embargo, es importante mencionar que estos resultados no deberían tomarse como una regla general debido a las limitaciones del protocolo diseñado e implementado.

Este documento propone un protocolo determinado al que se le podrían introducir varios cambios para obtener resultados mas precisos.

- En primer lugar, la característica empleada en todos los algoritmos para definir el *target* es el color. Esta implementación tiene muchas ventajas, pero obviamente, se podría utilizar un modelo mas complejo y analizar cómo afecta al comportamiento de los algoritmos.

- En segundo lugar, a pesar de que el dataset fue creado lo mas completo posible, es importante tener en cuenta que sólo se han analizado determinados problemas. Se podrían crear y añadir secuencias con importantes cambios de apariencia u otros problemas de tracking para tener un dataset más completo.

- A pesar de que el sistema se realizó lo mas automático posible, sería interesante optimizar el funcionamiento, especialmente si el dataset aumenta. Esto incluye (pero no está limitado) a la prueba de videos con cada algoritmo, la creación y evaluación de los ficheros de anotación para cada prueba y el análisis de dichos ficheros para determinar la precisión en cada caso.

- Sería interesante también comprobar los tiempos de ejecución de forma más exhaustiva, teniendo en cuenta por ejemplo cómo afecta el cambio de diferentes parámetros al consumo de tiempo .

- Por último, se ha observado que la definición de un buen comportamiento de un sistema de seguimiento no está demasiado clara, ya que un solapamiento espacial del 50% (entre la estimación del *target* y la anotación del *ground-truth*) es válida para algunas aplicaciones y no para otras. Por lo tanto, se podrían investigar nuevas métricas que no dependieran de la aplicación (como se detalla en [7]).

Appendix H

Presupuesto

1) Ejecución Material

- Compra de ordenador personal (Software incluido) — 2.000 €
- Alquiler de impresora láser durante 6 meses — 260 €
- Material de oficina — 150 €
- Total de ejecución material — 2.400 €

2) Gastos generales

- sobre Ejecución Material — 352 €

3) Beneficio Industrial

- sobre Ejecución Material — 132 €

4) Honorarios Proyecto

- 1800 horas a 15 €/ hora — 27000 €

5) Material fungible

- Gastos de impresión — 280 €
- Encuadernación — 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto

32.774 €

7) I.V.A. aplicable

- 16% Subtotal Presupuesto — 5.243,8 €

8) Total presupuesto

- Total Presupuesto — 38.017,8 €

Madrid, Febrero de 2012

El Ingeniero Jefe de Proyecto

Fdo.: Mónica Lozano Cruz

Ingeniero Superior de Telecomunicación

Appendix I

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un protocolo de evaluación de algoritmos de seguimiento de objeto (*tracking*). En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1) La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2) El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3) En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4) La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5) Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6) El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7) Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8) Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9) Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10) Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11) Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12) Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13) El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14) Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15) La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16) La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17) La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18) Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19) El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20) Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21) El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22) Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23) Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1) La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2) La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3) Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4) En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5) En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6) Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7) Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8) Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9) Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10) La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11) La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12) El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.